Simulation Modeling and Analytics of Human Decision Process and Segmentation of Population through Simulated Behavioral Data

by

Fan Xia

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computational Mathematics

Waterloo, Ontario, Canada, 2018

Abstract

Accurate segmentation of the general population allows companies to effectively allocate marketing resources and maximize opportunities. In this project, we were entrusted by an anonymous startup to simulate, segment and analyze a given designed context. In this research, firstly, we leverage multiple criteria decision-making theory, credibility theory, and prospect theory to present a complex and systematic simulation model that mimics human's process of decision making, reason generating and recommendation learning. Secondly, we design an accurate and fast approach to draw segmentation on the population through simulated behavioral data. As a result, our simulation model is general, flexible and competitive to cover most real-life situations, and our segmentation approach is efficient and reliable in different scenarios.

Acknowledgements

I would like to thank Jason who made this thesis possible. And great thank to Prof. Ben who gave a lot suggestions and help me finish this research together.

Dedication

This is dedicated to the one I love.

Table of Contents

List of Tables

List of Figures

Chapter 1

Problem setting

1.1 Motivation

Accurate segmentation of the general population into subgroups with different characteristics can be of immense value to the different industries. For example, for the video streaming companies like YouTube, segmenting the users into different groups based on their preferences would render the promotion of advertisements more efficient and fruitful. For e-commerce companies like Amazon, segregating clients based on their locations, helps to develop an efficient and smooth logistical administration. By targeting specific groups of customers, the segmentation model enables companies to effectively allocate marketing resources and take advantage of the opportunities. In this research, we propose to identify segments based on people's knowledge, expertise, and preferences, which not only assist companies to allocate resources, rather also helps them to analyze the characteristics of their customers. For example, people who prefer to exercise habitually are more physically healthier than others. While people having higher knowledge of fashion usually are good at dressing up. Our motivation is really about constructing a new approach to segment based on behavioral analysis and comparatives of an intuitive social game construct.

1.2 Background

This research had an industrial collaboration with an anonymous startup promoting healthy lifestyles through user participation. In particular, we mainly focus on an engaging in-APP game that is competitive in nature and is designed to incentive healthy behaviors.

The APP collected data on users activities through interactive participation, which was then analyzed to identify the users' self-awareness, knowledge and response to incentives. This analysis was utilized to segregate users in different segments or subgroups.

One of the main endeavors of this project was to design a simulation model that mimics a different user's decision making processes. This exercise was necessitated as the real data was not available. Moreover, as the segmentation process was dynamic, with people in different subgroup being treated differently, it could have led to a butterfly effect with arguably different results. The simulation model helped us formulate more experiments and design a better and dynamic simulation algorithm. This simulation model has been generic and flexible to cover the various types of potential users and situations. We used the simulation model to test:

1. The ability to identify human behavior patterns.

2. Demonstrate how incentives impacted the segment classifications.

3. Quantify the assimilation of cohort data that improves knowledge over a period of time.

4. Provide suggestions on how to classify members into just a handful of groups.

Once the simulation model was developed, the second focus of this project was to devise an analytical toolkit to analyze the user data. This toolkit was first tested using simulated data and has been envisioned to be applied to real data in the future.

Once the simulation model is developed, the second focus of this project is to develop an analytical toolkit to analyze users data. This toolkit will first be tested using simulated data and is envisioned to be applied to real data in the future.

1.3 Overview of methodology

1.3.1 Description of the game

High level

The mobile APP we were entrusted to analyze is a wellness APP It delivers healthy micro activities to an individual with the intent of maximizing the user's probability of good health and longevity. Users of this APP are frequently asked to upload photos of their healthy activities and evaluate with other image of healthy activities. After uploading the photos, they would receive an evaluation and recommendation from the in-house expert, which nudges them to become healthy.

Low level

In this research project, we mainly focused on the peer to peer wellness game, because we believed it contained the largest quantity of information among all the activities. As shown in Figure 1.1, in this game, the player was asked to provide the healthiest activity he did recently in the form of a photo. These photo moments were then put into a friendly competition and the photo owners provided with the evaluation and recommendation from the in-house expert.

After uploading the photos, players receive some points as an award. The number of points depends on the players decision, he can choose to gain a fixed number of point or enter into a bet. If he chooses to enter the bet, he can gain a higher number of points if he wins the food comparison and lower number of points if he loses the comparison.

Extra details

In the APP, after voting an activity pair, experts and players are asked to provide reasons for the vote, expert will also need to give a recommendation to players be voted. The reasons similarity, and the feedback of players from recommendation will also be analyzed as indicators of segmentation.

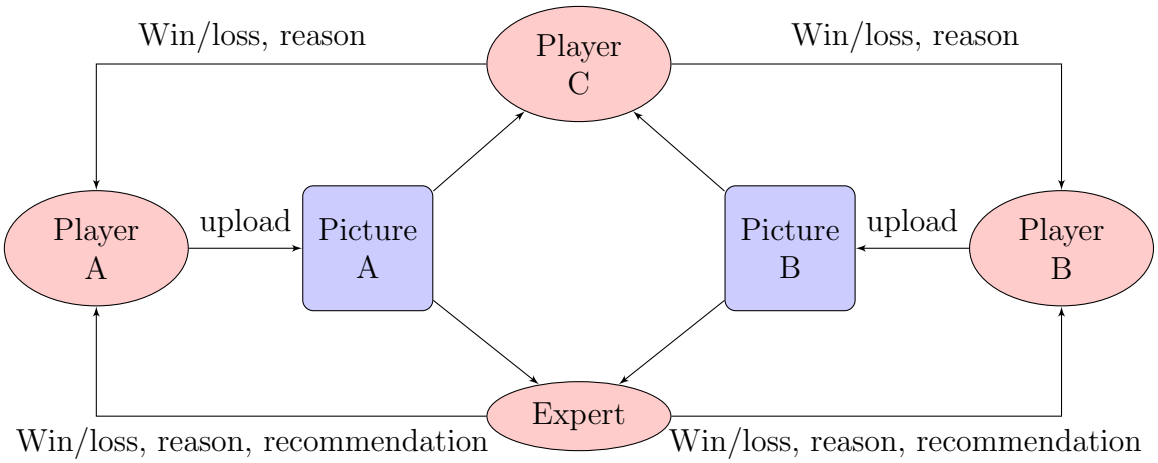


Figure 1.1: The process of the game

1.3.2 Related literature

As this research targets a specific circumstance, there were no preceding researches we could refer to, except for some partially relevant data to support our research. For the decision-making process of voting, people usually adopt different criteria. Zadeh[1965] introduced the fuzzy-set theory to simulate the decision-making process with multiple criteria. The main idea was to assign different scores to these multiple sets with numerous criteria.The score of a given object was equal to the score of the set which the object was most likely to be. Gass & Saaty[1955] proposed weighted sums system that multiplies each criterion with a positive weight and summed them up as one single criterion. Wierzbicki [1980] introduced the achievement scalarizing function which also combined the multiple criteria into a single criterion but in a different way. These decision-making model were all rational. However, most of them were not compatible with our research as our simulation was not just a pure decision-making model, we also accounted for reasons, recommendations and players' learning system for this purpose.

1.3.3 Preview of conclusion

Our simulation result illustrated that, the reason of difference or the agreement rate with the expert, are the most efficient features to detect expert-like players. The cumulative win rate was the best parameter for segmenting same level players. While the degree of reasoning difference was a good indicator for nudgable players.

1.4 Table of contents

Chapter 2 deals with the simulations for the entire gaming process, uploading activities, voting, providing reasons and recommendations from the prospective players and the experts, and how players learn from the experts. Chapter 3 elaborates how we used the credibility theory to update a person's confidence and how we leveraged the prospect theory to simulate the decision-making process when entering the bet. Chapter 4 interprets the segmentation model and the performance of different segmentation criteria under different scenarios. And, Chapter 5 summarizes our works and provides suggestions.

Chapter 2

Appreciation-Perception Based Decision Model

In this chapter, we will illustrate our simulation model for uploading activities, voting, and providing reasons and recommendations, from the prospects of players and the expert. We will start from introducing the simulation model of multiple criteria decision making process. Then we analyze the pros and cons of different decision models and explain the rationale behind our approach. Next, we illustrate how we define the decision model for the expert and players, and explain how they different from each other. Finally, we present our method to generate reason and recommendation based on different decision model, and how we simulate the learning process of players toward the expert's recommendation.

2.1 How people make decisions

Imagine that you are provided with two choices: going to the gym tonight, or attending a party with your friends, how will you choose? we know that, going to the gym is not so entertaining, but it is good for our health and personal attractiveness; attending a party is quite enjoyable and will improve your social skill, however, it can not improve your physical healthiness.

Zionts (1979) points out that, the diversity of decisions comes from the difference of criteria. In other words, different people have various evaluate criteria toward the potential profit of options, which leads to different choices. So we make two assumptions:

1. Faced with multiple choices, people will choose the one with highest score.
2. People have their own criteria to calculate the score of options.

2.2 Activity Space

To illustrate how we simulate the decision-making process, we start by introducing how we define activities mathematically.

In this chapter, we use the process of choosing food to eat as the main target. One reason is that choosing food is the most common decision we will make, and the process is quite similar to the decision-making process toward activities. And the other reason is using food instead of activity helps us convey our idea to readers. In this section, we firstly review some previous research about quantifying foods to support the rationality of our approach. And then we interpreter why we set some restrictions on our definition and why they make sense. In this paper, food is just a commonplace of healthy activity.

2.2.1 Relevant researche

Jolie M et.al [2009] use 12 nutritional variables to represent a given food, and use these features to calculate the nutrition score.

There is a website called Epicurious (https://www.epicurious.com) providing 330000+ recipes with ingredient and nutritional content.

Jaan Altosaar[2017] propose an idea of converting food into vector space using the embedding algorithm from Word2Vec. It is a feature learning technique in Natural Language Processing (NLP) that maps words into vectors of real number while keeping semantics relationship of the word. For example: King - man + woman = Queen. After training the embedding algorithm by 95,896 recipes, the model is capable to convert food into a vector representation of 100 dimensions as long as the name of the food is given and also in the training data. Similar food will have a close distance in food space.

2.2.2 Activity Space Definition

All of these studies proved that there are some methods which could convert food into a vector space with each dimension representing one property of the food, so are activities. Greco et.al[2016] in his book, < Multiple criteria decision analysis>, hypothesized that, every analyzable object can be represent in a vector space. For example, going to the gym could be labeled as “health”, “improve physique”, “not relaxed” and etc. Then we could create a vector space of “fitness”, “improvement”, and “degree of fatigue” to represent this activity.

Figure 1.1.a demonstrate how the distribution of food will be in the food space. For visualization purpose, we assume food properties has two dimensions: nutritive value and calorie. However, no matter how large the domain of the feature will be, we need to

normalize it to eliminate the effect of vary scale. In the simulation, the domain of each dimension is assumed to be [0,1).

$$F = (f_1, f_2, f_3, ..., f_n) \text{ where } f_i \in [0, 1) \text{ for } i \in [1, 2, .., n]$$

So in simulation, the distribution of foods in food space is actually the distribution of foods after normalization. Figure 1.1.b demonstrate how the distribution will be in the simulation.

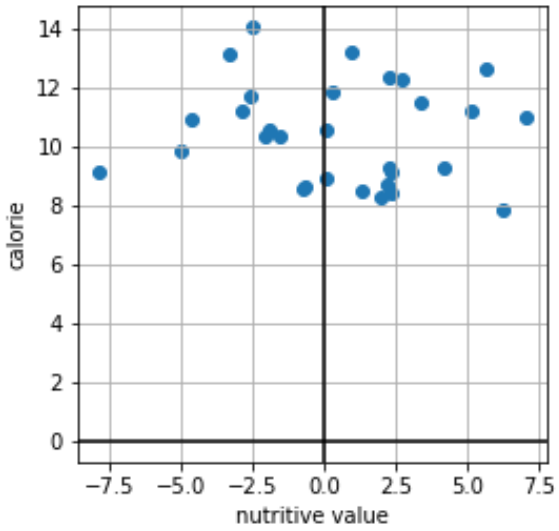


Figure 2.1: Food space

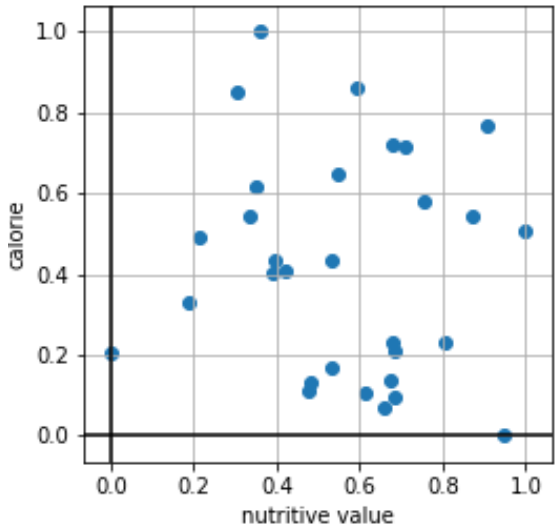


Figure 2.2: Food space with normalization

2.2.3 Dummy variable

Intuitively, Some features of food are hard to be measured, for example, the amount of Vitamin C or the level of fitness of going to the gym. Even for the expert, It is hard to tell how many Vitamin C the food contained. However, distinguishing whether the food contains Vitamin C is an easier job. So some dimensions of activities are set as dummy variables. The final definition of our Activity space is :

$$F = (f_1, f_2, f_3, ..., f_n)$$

where $f_i \in [0, 1)$ for $i \in [1, 2, .., m]$ and $f_i = 0$ or 1 for $i \in [m + 1, m + 2, .., n]$, $m \leq n$

2.3 Activity Score:

In this chapter, we illustrate how we simulate the scoring system of individuals. In the beginning, we introduce five possible metrics to calculate the total score (τ) of an activity. Each metric is designed to fix some drawback of the last one. Then we present the definition of appreciation and perception, and how we measure the knowledge and cognitive bias of individuals.

2.3.1 Piece wise Constant scoring system

The piece wise constant scoring system is widely used in the scoring topic. For each dimension of features, it firstly divides the domain of feature into several subdomains by some endpoints $(b_1, b_2, .., b_{m-1})$, then maps each subdomain into a predefined value $(s_1, s_2, ..., s_m)$. In the end, it sums up the value of each dimension together as the total score of the activity.

$$S_n = \begin{cases} s_{n,1} & 0 \leq f_n \leq b_{n,1} \\ s_{n,2} & b_{n,1} < f_n \leq b_{n,2} \\ ... & \\ s_{n,m} & b_{n,m-1} < f_n \leq 1 \end{cases}$$

$$\tau = \sum_{i=1}^{n} S_i$$

For example, Marjorie et.al [2002] use alternate healthy eating index (AHEI) score method to calculate the total score of food. The main idea is using predefined criteria to evaluate the quantities of different components, then each level of components quantity corresponding to an AHEI score. In the end, the total score of food was represented as the sum of all components AHEI score. This method is a piece wise constant scoring system.

This method is easy to use as it only takes the fuzzy value as input to calculate the total score.

However, there are some drawbacks of this system. Firstly, piece wise constant model usually contains more parameters than linear model, so it depends a lot on the size of data and is easily overfitting. Secondly, it is not suitable for the decision-making process. As it based on piece wise constant, some points in the activity space may have the same score while their Euclidean distance is quite large. This will result in no activity been chosen as they got the same score. For example, for a scoring system of:

$$S_1 = \begin{cases} 1 & 0 \leq f_n \leq 0.3 \\ 2 & 0.3 < f_n \leq 0.6 \\ 3 & 0.6 < f_n \leq 1 \end{cases} \quad \text{and} \quad S_2 = \begin{cases} 3 & 0 \leq f_n \leq 0.3 \\ 2 & 0.3 < f_n \leq 0.6 \\ 1 & 0.6 < f_n \leq 1 \end{cases}$$

$$\tau = S_1 + S_2$$

Figure 2.2 shows the distribution of points with the same score in the activity space. All the points share the same scores.

All the decision making models based on piece wise constant system all have the same problem. For example, the fuzzy set model present by Zadeh[1965]. This model defines some sets in the activity space, and assign a score to each set. The score of an activity equal to the score of the set where the activity is most likely to be in. So the score of activities will be the same if activities are all in the same set.

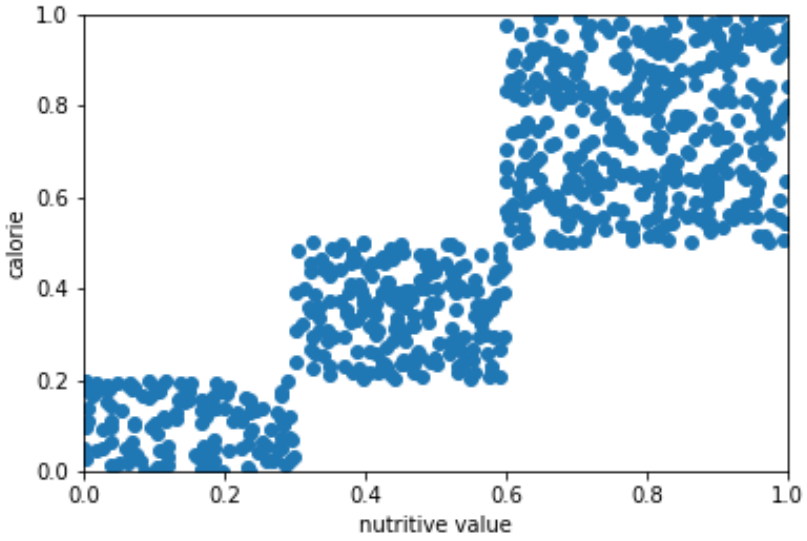


Figure 2.3: Example of points with same score in activity space

2.3.2 Weighted Sum scoring system

To fix the drawbacks of piece wise scoring system, we introduce the weighted sum scoring system. It assigns different weights $(w_1, w_2, .., w_n)$ to each features $(f_1, f_2, .., f_n)$ and calculates the weighted sum of features as the total score of activity.

Predictor variables	Coefficient
(Intercept)	0.710 (0.207)
Total fat(g)	-0.0538 (0.0414)
Saturated fat(g)	-0.423 (0.0944)
Cholesterol(mg)	-0.00398 (0.0033)
Sodium(mg)	-0.00254 (0.00044)
Total carbohydrate(g)	-0.03 (0.0110)
...	...

Table 2.1: Martin et.al. [2009]: Model for predicting food/beverage healthiness based on 12 nutritional variables

$$\tau = \sum_{i=1}^{n} w_i f_i$$

Martin et.al. [2009] asked some experts to rated each of the 205 foods as 11 scales, from -5 (very unhealthy) to 5 (very healthy). And they present a linear regression model fitted those date to predict the healthiness of other foods. The result is shown in Table 2.1.

This method solves the "same score" problem of piecewise constant method. Also, the linear model has fewer parameters compared with the last one. It is a good choice when the dataset is small, and usually, activity score data is hard to collect as the expert resource is expensive and scarce.

The drawback of the weighted sum scoring system is that it can't take features interaction into consideration. For example, Vitamin D could boost the absorption of calcium, so it is healthier to eat food containing vitamin D and calcium together. The current model is not capable to simulate the interaction.

2.3.3 Weighted Sum scoring system with Interaction

To address the problem of no interaction. We expand the total score function:

$$\tau = \sum_{i=1}^{n} w_i f_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \bar{w}_{ij} f_i f_j$$

where $\bar{w}_{ij} = 0$ if f_i has no interaction with f_j

To not make the model too complex, we only consider the interaction up to two features. As for the example of Vitamin D and calcium, the total score in this model will be higher

the last model's as $\bar{w}_{vitamin_D,calcium} > 0$

The shortage of this model is that the partial derivative of total score with respect to each feature is positive. It means that as long as the weight of the feature is positive, the higher amount of feature is always better. However, in reality, sometimes the most appropriate amount is the best.

2.3.4 Weighted Sum scoring system with Interaction and Kernel Function

we use the kernel function ($k(f)$) to address the monotonicity problem. we set the appropriate amount to be $(a_1, a_2, .., a_n)$. And assume that:

1.

$$\tau = \sum_{i=1}^{n} w_i \bar{f}_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \bar{w}_{ij} f_i f_j$$

$$\bar{f}_i = k(f_i)$$

2. The total score is maximized when $(f_1, f_2, .., f_n) = (a_1, a_2, .., a_n)$:
$\forall (f_1, f_2, .., f_n) \neq (a_1, a_2, .., a_n)$

$$\tau((a_1, a_2, .., a_n)) \geq \tau((f_1, f_2, .., f_n))$$

3. The total score function may be symmetric around $(a_1, a_2, .., a_n)$:
$\forall x > 0$ and $i \in [1 : n]$,if $a_i + x < 1$ and $a_i - x > 0$,

$$\tau((f_1, .., a_i - x, .., f_n)) = \tau((f_1, .., a_i + x, .., f_n))$$

Based on the assumption, we introduce two kernel function: piece wise linear and the quadratic function.

Piece wise linear kernel function:

$$k(f_i) = \begin{cases} \frac{f_i}{a_i} & 0 \leq f_i \leq a_i \\ -\frac{f_i}{a_i} + 2 & a_i < f_n \leq 1 \end{cases}$$

Quadratic kernel function:

$$k(f_i) = -\frac{f_i^2}{pcp_i^2} + 2\frac{f_i}{pcp_i}$$

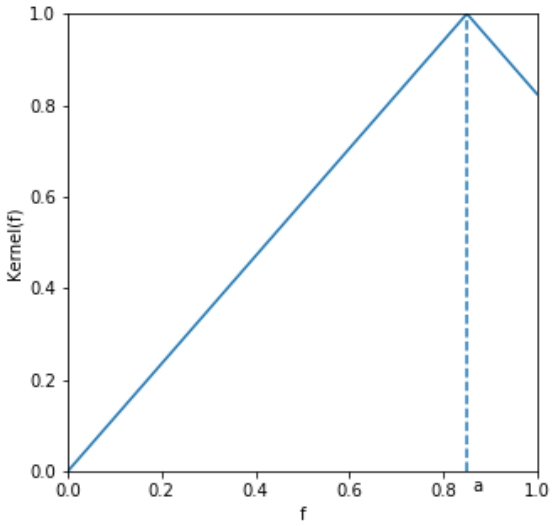


Figure 2.4: Piece wise linear kernel function with appropriate amount = a

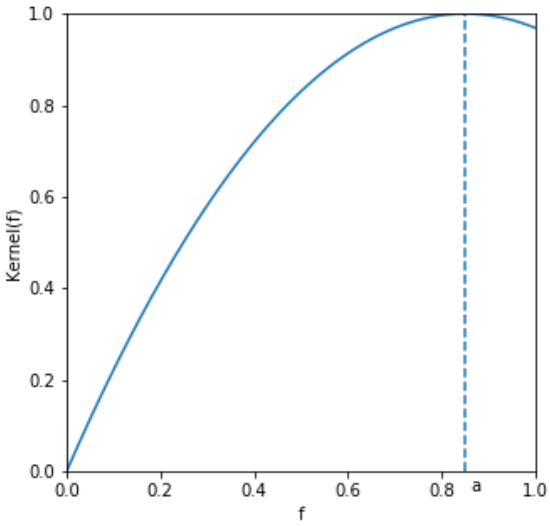


Figure 2.5: Quadratic kernel function with appropriate amount = a

By using the kernel function, $\sum_{i=1}^{n} w_i \bar{f}_i$ take the maximum value only when $(f_1, f_2, .., f_n) = (a_1, a_2, .., a_n)$, which means every feature research its appropriate amount. If there is no demand for appropriate amount, then $a_i = 1 \ \forall i \in [1 : n]$

2.4 Appreciation and Perception

As we have defined how the activity score is calculated, in this section, we want to explain why and how people have different preference toward the same pair of activities and how we simulate it by introducing appreciation, and perception. In this section, we used weighted sum scoring system with interaction as our score system.

There is another black box simulation model which use XGBoost and decision tree model to simulate the process, which will be illustrate in the appendix as bonus part.

In weighted sum scoring system with interaction, there are three important variables that control the final score of an activity, f_i, a_i and w_i. In our simulation, we called f_i perception, a_i appropriate amount, and w_i the appreciation. In this section, we first present how we define knowledge of a person, then we introduce how to represent cognitive bias.

2.4.1 Appreciation

We hypothesis that the expert is the "master" and has "omni-potent". So the experts' appreciations are always the same, which is $(w_{1e}, ..., w_{ne}, \bar{w}_{11e}, ..., \bar{w}_{nne})$. And different from experts, ordinary people's appreciation may bias from the expert's and different from each others. Many research (Jennifer et.al[2015],OLGA[2016],Sehar[2017]) points out that, people's appreciations are conditional. Different circumstance and emotion state will lead to diverse appreciations. As we can not track the circumstance and emotion state of a person, we consider appreciations are random variables and not consistent across time.

We use consistency $(cns_1, ..., cns_n, c\bar{n}s_{1,2}, ..., c\bar{n}s_{n-1,n})$ to describe the uncertainty of a person's appreciation, and use knowledge $(k_1, ..., k_n, \bar{k}_{1,2}, ..., \bar{k}_{n-1,n})$ to measure how far away his mean appreciation $(w_{m1}, ..., w_{mn}, \bar{w}_{m,1,2}, ..., \bar{w}_{m,n-1,n})$ is biased from expert's appreciation $(w_{1e}, ..., w_{ne}, \bar{w}_{1,2,e}, ..., \bar{w}_{n-1,n,e})$

If one player has high *consistency* in some dimension of activity space, his appreciation of this dimension should be very consistent across time. If one player has low *consistency* in some dimension of activity space, his appreciation should be quite divergent. Figure 2.6 shows the difference. In simulation, we set:

$$\forall i \in [1, .., n, 11, .., nn], \quad w_i \sim N(w_{mi}, w_{mi}^2(1 - cns_i)), \quad cns_i \in [0, 1]$$

where $(w_{m1}, ..., w_{mn}, \bar{w}_{m11}, ..., \bar{w}_{mnn})$ is the mean appreciation of this person

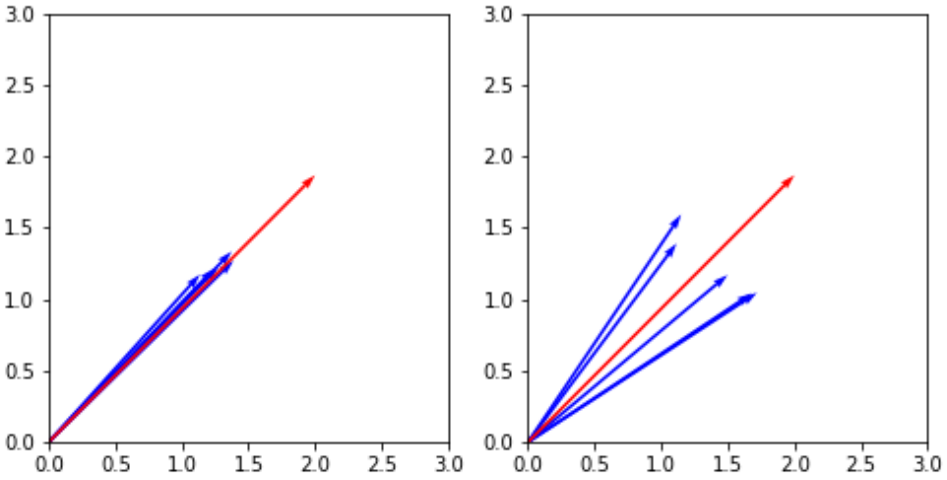


Figure 2.6: Left: The long vector represents mean appreciation, short vectors represents 5 appreciations of the same person with high consistency. Right: Appreciation of the low consistency people.

In the simulation, we set overall knowledge (kl) is the cosine similarity between the person's mean appreciation and expert's:

$$kl = cos(\theta) = \frac{\sum_{i=1} w_{mi} * w_{ei}}{\sqrt{\sum_{i=1} w_{mi}^2}\sqrt{\sum_{i=1} w_{ei}^2}}$$

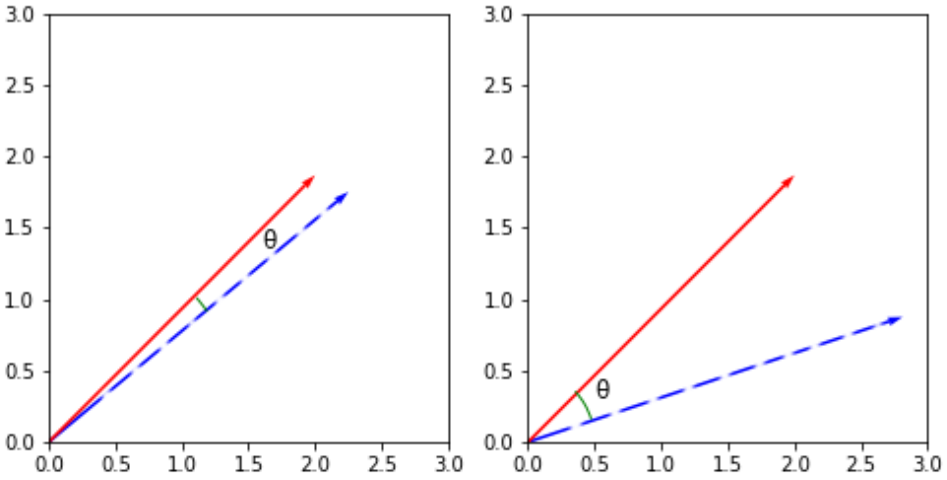


Figure 2.7: Left: The Solid vector represents expert's appreciation, the dashed vector represent a high knowledge people's mean appreciation. Right: Appreciation of the low knowledge people.

As $kl = cos(\theta)$, so $kl \in [-1, 1]$, when $kl = 1$, this person's appreciation is same as the expert's; when $kl = -1$, his appreciation is the exactly opposite to the expert's; when $kl = 0$, his appreciation has no correlation with expert's. Figure 2.7 the difference between high knowledge person and low knowledge person.

To separate the overall knowledge to several dimension, we set $(k_1, ..., k_n, \bar{k}_{1,2}, ..., \bar{k}_{n-1,n})$ as measurement the knowledge bias in each dimension. It is not appropriate to compare w_i with w_{ei} directly. Because people with the different appreciation may still make the same decision, as long as $w = aw_e, a > 0$. To solve this problem, all the appreciations needs to be normalized before comparing.

$$k_i = \frac{w_{mi}}{\sqrt{\sum_{i=1}^{n} w_{mi}^2}} - \frac{w_{ei}}{\sqrt{\sum_{i=1}^{n} w_{ei}^2}}$$

Notice that, as we defined our total score function as:

$$\tau = \sum_{i=1}^{n} w_i k_i(f_i) + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \bar{w}_{ij} f_i f_j$$

The kernel functions $k_i(x)$ are also key parameters to evaluate a persons knowledge. Given the expert's appropriate points $(a_{e1}, a_{e2}, ..., a_{en})$ and the player's appropriate points $(a_1, a_2, ..., a_n)$, we define the modified overall knowledge $\bar{kl}$ as:

$$\bar{kl} = kl(1 - 2\frac{\sum_{i=1}^{n}(a_{ei} - a_i)}{n})$$

When $a_{ei} = ai, \forall i \in [1:n]$, $\bar{kl} = kl$, and when $a_{ei} - ai = 1, \bar{kl} = -kl, \forall i \in [1:n]$.

In the simulation, we use following step to sample a person's mean appreciation:

Given modified knowledge $\bar{kl}$ and consistency $(cns_1, .., cns_n, cns_{11}, .., cns_{nn})$, we will first sample w_{mi} from $N(w_{ei}, w_{ei}^2 kl)$, then calculate the $cos(\theta)(1 - 2\frac{\sum_{i=1}^{n}(a_{ei}-a_i)}{n})$ as the true knowledge, if true knowledge $\notin [\bar{kl} - 0.1, \bar{kl} + 0.1]$, resample until it is in the interval. Every time when calculate the score, sample w_i from $N(w_{mi}, w_{mi}^2(1 - cns_i))$

2.4.2 Perception

As we have defined what is the appreciation $(w_1, ..., w_n, \bar{w}_{1,2}, ..., \bar{w}_{n-1,n})$ in our simulation model, the other key value that will determine the activity score is $(f_1, f_2, ..., f_n)$. It represents people's views toward an activity, We call it perception.

People's perception of the same activity may be different. For example, there are so many people smoke in their daily life. They choose smoking not only because the entertainment received from smoking is higher than the harm, but also they are not aware of how large the harm would be. That why many anti-smoking ads try to help people prevent smoking by showing them the harm: reinforcing the perception of smoking harm.

In 2017 food & health survey by International Food information Council foundation, it is found that only 13% of people surveyed know exactly the nutrition ingredient of the food they eat. This survey indicated that perception is also a key point to distinguish nutrition expert from normal people.

There are lots of cognitive bias in psychology and behavior economics. These cognitive biases follow some rules, for example, IKEA effect. IKEA effect represents the tendency that people like to put a higher value on objects that they partially or entirely assembled themselves.

Those kinds of bias usually follow some rules, but it is hard to track. For those biases, instead of tracking all the reason, we prefer to consider them as a random variable.

Expert-like person will has low bias on the perception of activities $(p_1, p_2, .., p_n)$. We use perceptivity $(pcp_1, pcp_2, .., pcp_n) \in [0, 1]^n$ to represent the randomness of the bias.

The bias may have such properties:

1. Like activity space, $\forall i \in [1, 2, .., n]$, $p_i \in [0, 1]$
2. Given an activity $(f_1, f_2, .., f_n)$,$\forall i \in [1, 2, .., n]$, if $pcp_i = 1$, $p_i = f_i$
3. If $pcp_i = 0$, the value of p_i should be very random

So we set perception:

$$\forall i \in [1, 2, .., n], p_i \sim pcp_i f_i + (1 - pcp_i)U(0, 1)$$

2.4.3 Simulation process

After introducing both appreciation and perception, in this section, we will summarize the process of calculating the score of an activity.

Given an activity $(f_1, f_2, .., f_n)$, we first calculate the person's perception $(p_1, p_2, .., p_n)$ toward the activity by $p_i \sim pcp_i f_i + (1 - pcp_i)U(0, 1)$,then map the perception$(p_1, p_2, .., p_n)$ into $R^{\frac{(n-1)n}{2}}$ space: $(p_1, .., p_n, p_1p_2, .., p_{n-1}p_n)$

Next, sample appreciation $(w_1, ..., w_n, \bar{w}_{1,2}, ..., \bar{w}_{n-1,n})$ from $N(w_{mi}, w_{mi}^2(1 - cns_i))$, calculate the score of the activity by $\sum_{i=1}^{n} w_i p_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \bar{w}_{ij} p_i p_j$

Notice in our simulation, the score for the same activity may different from each time because of the randomness of perception.

2.5 Upload Best Activity

In the app we were entrusted to simulate, every players is asked to upload the healthiest activity he did recently. As we have defined how the activity score is calculated by each individual, in this section, we present how we simulate this process using the scoring system we defined.

2.5.1 Score based activity selection

The idea is that for every player, he will first generate several activities randomly, then pick the one that has the highest score to him as the activity he uploads. We set nof as the number of food the healthiest food is picked from. For clear notation, we set the sample set of activities as Φ, the final choice of the activity as F.

$$\Phi = (F_1, F_2, .., F_{nof})$$

where $F_j = (f_{j1}, f_{j2}, .., f_{jn})$ and $f_{ji} \sim U(0,1), \forall i \in [1,2,..,n], \forall j \in [1,2,..,nof]$

$$F = argmax_{F_i}(\tau(F_i))$$

2.5.2 Healthy activity distribution

In this section, we will present the distribution of the best activity given different parameters.

The nof is set as 14. For visualization purpose, expert's appreciation is set as (1,1).

Figure 2.8 presents the best activity distribution of a high-level knowledge and perceptivity person. Figure 2.8.a is the activity distribution of the expert, his activity is almost on the top right corner, which is the same direction as his appreciation (1,1). We can see with the consistency get lower, people's best activity spread to the other corners and edges of the activity space. This kind of persons have the impulsion to eat healthy food, and also know what is good. However, they are not very confident about their knowledge.

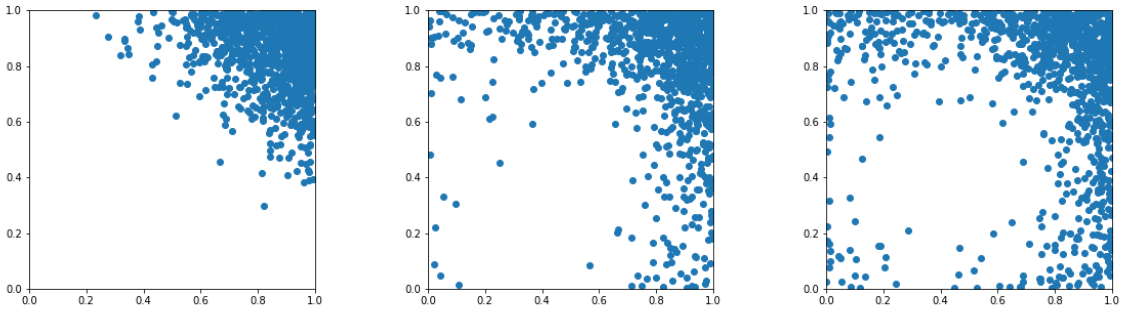
Figure 2.9 presents the best activity distribution of a high consistency and perceptivity person. Different from Figure 2.8, with knowledge decreased, best activities move to the other sides because their knowledge about the food activity is wrong. However, they are very confident about their knowledge.

Figure 2.10.a and 2.10.b present the best activity distribution of a low perceptivity person. We can see with perceptivity become lower, his best activities become more scattered. This kind of person may not have a strong willingness to improve themselves and easy to be influenced by cognitive bias.

Figure 2.10.c is what we believe a normal person's distribution: has some bias on the knowledge, not very confident of his own knowledge, and sometimes have some cognitive bias on the activity.

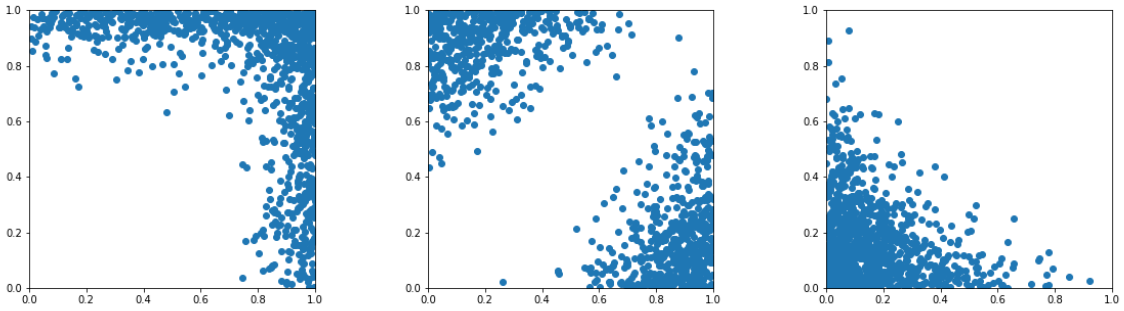
2.6 Recommendation & Reason

In the app, after voting an activity pair, experts and players are asked to provide reasons for the vote, experts will also need to give a recommendation to players be voted. In this section, we present how we quantify recommendation and reason given by expert and players. We use the recommendation sample from the mobile app to illustrate the rationality behind our approach.



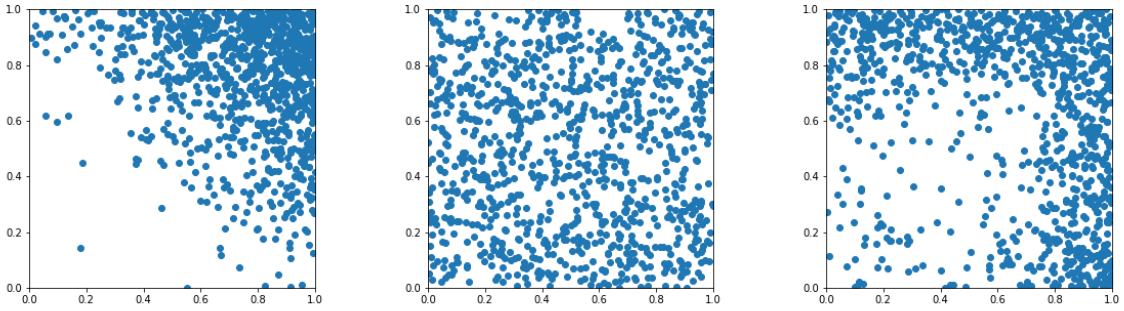
(a) *consistency* $= (1, 1)$ (b) *consistency* $= (0.5, 0.5)$ (c) *consistency* $= (0, 0)$

Figure 2.8: Best activity distribution with $w_e = (1, 1)$, *knowledge* $= 1$, *perceptivity* $= (1, 1)$



(a) knowledge=0.6 (b) knowledge=0.2 (c) knowledge=-1

Figure 2.9: Best activity distribution with $w_e = (1, 1)$, *consistency* $= (1, 1)$, *perceptivity* $= (1, 1)$



(a) perceptivity=(0.5,0.5) (b) perceptivity=(0,0) (c) Normal People

Figure 2.10: a,b: Best activity distribution with $w_e = (1, 1)$, *consistency* $= (1, 1)$, *knowledge* $= 1$ c: Best activity distribution with $w_e = (1, 1)$, *consistency* $= (0.6, 0.6)$, *perceptivity* $= (0.6, 0.6)$, *knowledge* $= 0.6$

2.6.1 Reason

When an expert or player vote a food pair, he also needs to provide the reason for the vote. The similarity between reasons from the expert and players is also a key value to evaluate the level of the player. By level, we means the higher level a player is, the more likely this player is an expert.

If the reason is given by the form of sentence, there are already some Nature Language Process techniques that measure the distance between sentence, such as Word Mover's distance, a technique that able to calculate the semantic dissimilarity between sentences like "how are you" and "how old are you".

While in the app, the job is easier as players and experts only need to choose from several predefined options. So the key point is calculating the similarity between the chosen options.

The number of options should be as few as possible to make the app more user-friendly. So we assume the number of options equal to the number of dimensions of activity space. In our simulation, we don't consider reasons of interactions as there is a limitation from the number of predefined options. Even there may be an interaction between features, experts and people can't express it by only choosing options.

As people can only submit their reason by choosing the predefined options, so the reason r in our simulation can only contain the information of "is the reason for winning" and "is not the reason for winning".

Therefor, in simulation, given activity 1: $F_1 = (f_{11}, f_{12}, .., f1n)$, activity 2: $F_2 = (f_{11}, f_{12}, .., f1n)$, the person's appreciation $(w_1, ..., w_n, \bar{w}_{1,2}, ..., \bar{w}_{n-1,n})$, and a threshold $\lambda > 0$, the reason $R(F_1, F_2) = (r_1, r_2, .., r_n, r_n)$ is calculated by:

if $\tau(F_1) > \tau(F_2)$:

$\forall i \in [1, 2, ..n]$, $r_i = 1$ if $w_i(f_{1i} - f_{2i}) > \lambda$, $r_i = 0$ otherwise

if $\tau(F_1) < \tau(F_2)$:

$\forall i \in [1, 2, ..n]$, $r_i = 1$ if $w_i(f_{1i} - f_{2i}) < -\lambda$, $r_i = 0$ otherwise

Then the similarity between the expert's reason $(r_{e1}, r_{e2}, .., r_{en})$ and the person's reason $(r_1, r_2, .., r_n, r_n)$ is:

$$\sum_{i=1}^{n} |r_{ei} - r_i|$$

However, there is a problem of the metric of similarity. We believe it is a pretty bad situation if the player gave the same reason as expert while they provide opposite vote. In the current metric, the similarity of this situation is the same of the situation when the similarity of reason and vote are all the same. We believe different pairs of reason and vote corresponding to different levels:

Reason agreed	Vote consistency	Level
Same reason	Same vote	Good
Different reason	Same vote	Fine
Different reason	Different vote	Bad
Same reason	Different vote	Very bad

Our current similarity metric cannot achieve this. It is because current reason does not contain the information of direction. So we modify the definition of reason as:

if $\tau(F_1) > \tau(F_2)$:

$\forall i \in [1, 2, ..n]$, $r_i = 1$ if $w_i(f_{1i} - f_{2i}) > \lambda$, $r_i = 0$ otherwise

if $\tau(F_1) < \tau(F_2)$:

$\forall i \in [1, 2, ..n]$, $r_i = -1$ if $w_i(f_{1i} - f_{2i}) < -\lambda$, $r_i = 0$ otherwise

The similarity is:

If $Vote_e(F_1, F_2) = Vote(F_1, F_2)$:

$$0.8 * \sum_{i=1}^{n} |r_{ei} - r_i|$$

If $Vote_e(F_1, F_2) \neq Vote(F_1, F_2)$:

$$1.2 * \sum_{i=1}^{n} |r_{ei} - r_i|$$

Table 3.1 shows the difference between new similarity score and old similarity score. We can see that our new measure follows our expectation and the old one is not.

Table 2.2: New similarity VS Old similarity

Reason agreed	Vote consistency	Level	Old similarity	New similarity
Same reason	Same vote	Good	0	0
Different reason	Same vote	Fine	1	0.8
Different reason	Different vote	Bad	1	1.2
Same reason	Different vote	Very bad	0	2.4

2.6.2 Recommendation

After voting, the expert will also provide recommendations to each player. The recommendation will help players to restructure their knowledge.

We assumed the number of recommendations equal to the number of expert's appreciation $(w_{1e}, ..., w_{ne}, \bar{w}_{11e}, ..., \bar{w}_{nne})$. The expert will provide recommendations to players according to features that contribute less score in activity comparison. For example, given player A's food $(f_{11}, f_{12}, .., f_{1n})$ and player B's food $(f_{21}, f_{22}, .., f_{2n})$, he will give A recommendation on feature i that satisfied $w_{ei}f_{1i} < w_{ei}f_{2i}$.

Because recommendations are sentence like "Try to eat more vegetables", and "Try to avoid oil food", the recommendations should be a dummy variable, with +1 representing encourage and -1 representing discourage. The sign of recommendation should be the same as the partial derivative of features.

Given activity 1: $F_1 = (f_{11}, f_{12}, .., f_{1n})$, activity 2: $F_2 = (f_{21}, f_{22}, .., f_{2n})$, the expert's total score function: $\tau(F)$ and a threshold: $\lambda > 0$, the recommendation $(rcm_1, .., rcm_n, rcm_{1,2}, .., rcm_{n-1,n})$ for player 1 is calculated by:

$$\forall i \in [1, 2, ..n],\ rcd_i = sign(\tfrac{\partial \tau_e(F)}{\partial f_i}) \text{ if } w_{ei}(f_{1i} - f_{2i}) < -\lambda,\ r_i = 0 \text{ otherwise}$$

In our score system, given the expert's appreciation: $(w_{1e}, ..., w_{ne}, \bar{w}_{11e}, ..., \bar{w}_{nne})$, appropriate points: $(a_1, a_2, .., a_n)$, we have:

$$sign(\frac{\partial \tau_e(F)}{\partial f_i}) = sign(w_i(a_i - f_i)), \forall i \in [1 : n]$$

$$sign(\frac{\partial \tau_e(F)}{\partial f_i j}) = sign(\bar{w}_{ij}), \forall i, j \in [1 : n], i < j$$

Learn from recommendation

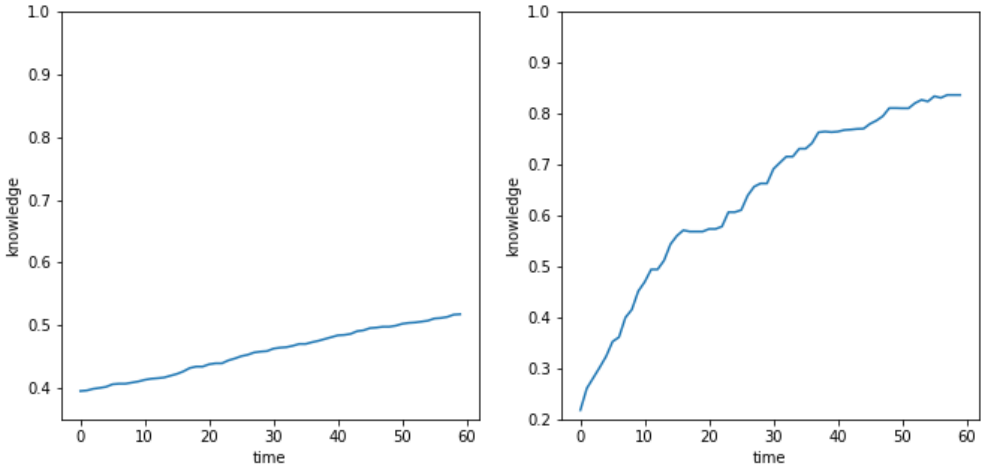
As we have defined how the expert generates recommendations, here let us talk about how people react to the recommendation.

Every individual has his own learning rate (lr). By learning rate, we mean the degree of willingness a person would like to follow expert's recommendation.

So given expert's recommendation $(rcm_1, .., rcm_n, rcm_{1,2}, .., rcm_{n-1,n})$ and learning rate (lr), the person will update their knowledge by adding the recommendation to his mean appreciation $(w_{m1}, ..., w_{mn}, \bar{w}_{m,1,2}, ..., \bar{w}_{m,n-1,n})$:

$$w_{mi} = w_{mi} + rcm_i lr, \forall i \in [1, .., n, (1, 2), .., (n-1, n)]$$

In the simulation, the lr is set as a constant. The benefit for such setting is that with time goes, as people attend more and more comparisons, his total appreciation $\sum_{i=1}^{n} a_{mi}$ is also generally increased. As lr is a constant, the effect of recommendation will decrease



(a) Learning rate=0.01 (b) Learning rate=0.1

Figure 2.11: Knowledge changed across time with different learning rate

if his confidence is stable. We believe the decreased influence of recommendation follows the reality.

Figure 2.11(a),2.11(b),2.12(a) show changes of knowledge of a low knowledge player with different learning rate. We can see that, the higher learning rate a person has, the faster his knowledge become similar to the expert's. Figure 2.12(b) shows the change of knowledge if we normalize mean appreciation w_m before updating. Different from our design, the effect of recommendation will not decrease as the weight is normalized before updating. We can see from figure 2.12(b) that, in this design, the knowledge of the player is not stable.

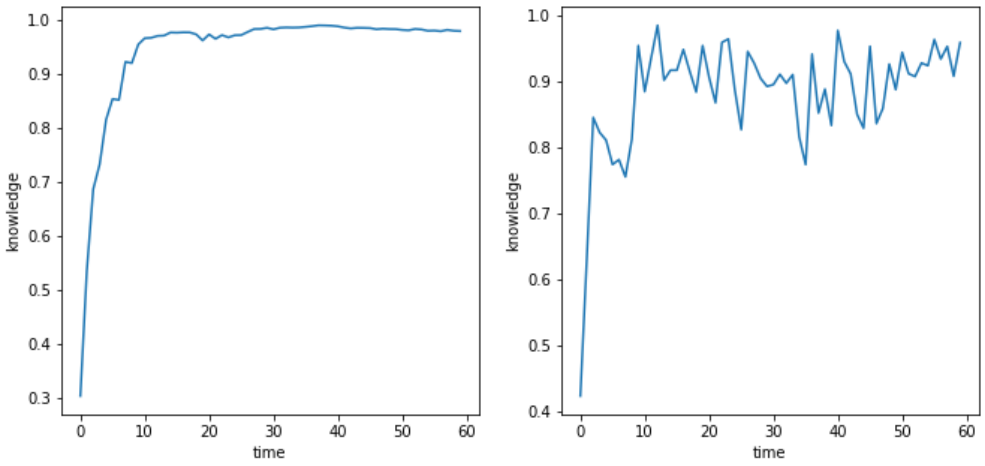
consistency & perceptivity

We assume with listening more and more from experts, players' consistency (cns) and perceptivity (pcp) will also increase.

So we define:

$$cns_i = cns_i + (1 - cns_i)2lrU(0, 1)$$

$$pcp_i = pcp_i + (1 - pcp_i)2lrU(0, 1)$$



(a) Our design

(b) Mean appreciation w_m is normalized before updating

Figure 2.12: Knowledge changed across time, with learning rate = 1

Chapter 3

Prospect-Credibility theory Based Decision Model

In the last chapter, we have illustrated how we simulate the decision-making process of choosing the activity. In this chapter, we will present another simulation model targeting a different decision-making process. We will use credibility theory and prospect theory to build this simulation model.

In the 'healthy food comparison' game, after uploading food photo and voting on other food pairs, players would receive some point as an award. The number of point a player could earn is based on his own decision: he can choose whether to gain a fixed number of point or enter into a bet. If he chooses to enter into a bet, he will gain a higher amount of point if he wins the food comparison and gain a lower amount of point or lose some point if he loses the food comparison.

In this chapter, we will present how we simulate the decision-making process using credibility theory and prospect theory. We will start by illustrating the confidence of a person and how confidence changes. Then present the decision-making process using prospect theory.

3.1 Confidence

We define confidence (C) represents the probability a person thinks he could win the current comparison. Every people has his own confidence and strength of the belief. In this section, we will first introduce the credibility theory, and then present how we simulate confidence using the Buhlmann credibility theory.

3.1.1 Credibility theory

Credibility theory is a form of statistical inference that combine several estimates to a more accurate estimate based on credibility of these estimates. If given two estimates x and y, by credibility theory, the new estimate is calculate by:

$$wx + (1 - w)y, 0 \leq w \leq 1$$

w is called the credibility factor. It is derived with the purpose of maximizing the likelihood, which would minimize the error of the estimate. The core idea of credibility theory is the weighted average. It sums up different estimates of a quantity based on the credibility of the estimates. Higher weight is given to more credible estimate. There are several way to calculate the credibility factor, in our simulation, we used Buhlmann credibility in updating the confidence.

Buhlmann credibility

Buhlmann credibility is the most popular form of credibility theory. It is also called least squared credibility or Bayesian credibility. It is calculated by minimizing the expected squared error of a quantity.

Suppose there are two estimates x and y with different mean squared errors u and v. Assume weight average is:

$$\alpha = wx + (1 - w)y, w \in [0, 1]$$

Then the expected squared error of α is:

$$MSE(\alpha) = w^2u + (1 - w)^2v$$

Take derivative of $MSE(\alpha)$ we could get:

$$\frac{dMSE(a)}{dw} = 2wu - 2(1 - w)v$$

Set the derivative equal to 0, it can be calculated that:

$$w = \frac{v}{u + v}$$

So, given estimates x and y, with mean squared errors u and v, the least square credibility estimate is calculated as:

$$a = \frac{vx}{u + v} + \frac{uy}{u + v}$$

The credibility factor w could be expressed as:

$$w = \frac{v}{u+v} = \frac{1}{1+\frac{u}{v}}$$

So $\frac{u}{v} \to 0, w \to 1$ and $\frac{u}{v} \to \infty, w \to 0$. This means the smaller mean squared error one estimate has, the higher weight is assigned to this estimate.

3.1.2 Least square credibility for confidence update

In our simulation, every person has two estimates of his expected win rate: his own belief b and the win rate wr calculated from history.

Because both the expert and players will vote on food pairs, every player will have two win rate: one is the win rate based on the expert's vote wr_e, the other is the win rate based on players' vote wr_p. We assume both wr_e and wr_p follow Bernoulli distribution, so they have mean $= wr$ and mean squared error $= \frac{wr(1-wr)}{n-1}$

We assumed the total win rate is a weight average of these two win rate:

$$wr = \frac{2wr_e + wr_p}{3}$$

Then the mean squared error of wr is:

$$MSE(wr) = (\frac{2}{3})^2 \frac{wr_e(1-wr_e)}{n_e - 1} + (\frac{1}{3})^2 \frac{wr_p(1-wr_p)}{n_p - 1}$$

We assume one's belief b is an estimate with mean square error = s. So one's confidence (C) is calculated by credibility theory:

$$C = \frac{MSE(wr)b + swr}{s + MSE(wr)} = wb + (1-w)wr$$

where

$$w = \frac{1}{1+\frac{s}{MSE(wr)}}$$

We can see if $\frac{s}{wr} \to 0$, $w \to 1$, then $C \equiv b$. If $n_e, n_p \to \infty$, then $MSE(wr) \to 0$, so $\frac{s}{MSE(wr)} \to \infty$, which indicates that $w \to 0$, as the result $C \to wr$.

As we can see here, s control the speed of update with confidence. We called s: susceptibility, and it measures the strength of someone's belief. The lower susceptibility someone has the less influence of win rate on his confidence.

3.1.3 Update rule

As we have defined the method to update the confidence, there are two options coming up about how to treat the initial belief:

1. Update initial belief and susceptibility with new confidence and mean squared error every time confidence is updated.

2. Keep initial belief and susceptibility unchanged across time

In our simulation, we treat initial belief as the second option. In this section, we will illustrate the difference between two options and gave reasons for our choice.

Option 1 process

If using option 1, the process of updating confidence will be:

At time t, the person has his own belief b_t and susceptibility s_t. After a certain time period tp, he collects $n_{e,t:t+tp}$ vote result from the expert and $n_{p,t:t+tp}$ vote results from players within this time period. As we have introduced in the last section, we will use these results within this time period as an evaluate, the person's confidence is calculated as:

$$confidence = wb + (1 - w)wr_{t:t+tp}$$

where

$$w = \frac{1}{1 + \frac{s}{MSE(wr_{t:t+tp})}}$$

The mean sqaured error of this confidence is:

$$MSE(C) = w^2 s + (1 - w)^2 MSE(wr_{t:t+tp})$$

Then we will update the brief and susceptibility with

$$b_{t+p} = C$$

$$s_{t+p} = MSE(C)$$

At the end of the next time period, we will use b_{t+p} and s_{t+P} as the brief and susceptibility to update the next confidence.

Option 2 process

If using option 2, the process of updating confidence will be:

Keep initial belief and susceptibility unchanged. At any time t, we use $n_{e,0:t}$ vote results from the expert and $n_{p,0:t}$ vote results from players across the time as an estimate. Calculate the confidence with the same rule:

$$c = wb + (1 - w)wr_{0:t}$$

where

$$w = \frac{1}{1 + \frac{s}{MSE(wr_{0:t})}}$$

Options difference

We set up a simulation to test the performance of two options. This is a simulation with 5 high-level players, 5 low-level players, and 15 normal players. By what we mean high, low and normal, Table 4.1 shows the parameter setting with different level. The detailed process of the simulation run is not necessary for this section, we will illustrate it in the next chapter. Every player's initial belief and susceptibility are set as random numbers sampled from U(0,1).

Figure 3.1 shows plots of the difference between players' confidence and win rate across time with option 1 and option 2. Generally, most people's confidence converges to his win rate. Option 2 has a better performance than option1 as everyone's difference shrink to [-0.05,0.05] in option 1, while the other is mostly between [-0.2,0.1] and has some outliers that the confidence diverges from win rate with time goes.

We draw figure 3.2 to figure our the reason for the divergence. Figure 3.2 present the time series of players' confidence with different options. We can see that in Figure 3.2.a, which is the figure for option 1, there are some players' confidence stay unchanged at 0. That is because, by chance, there may be a time period that someone's win rate is 0, which leads to:

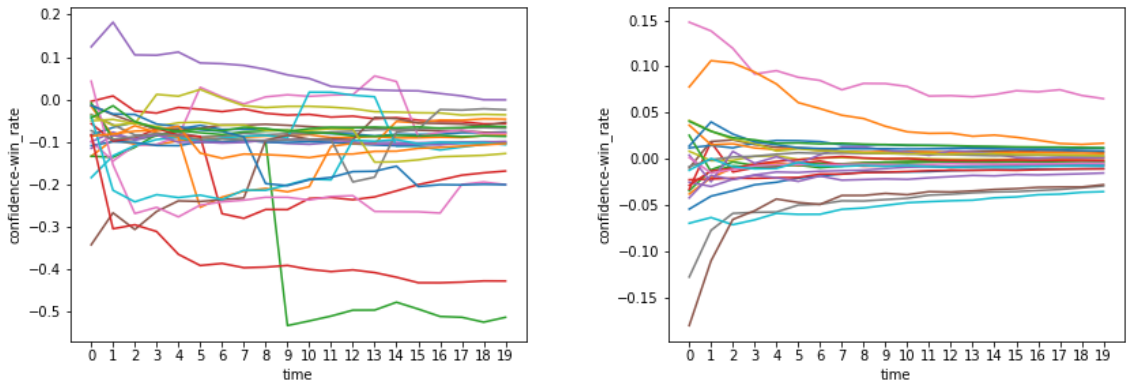
$$MSE(wr_{t:t+tp}) = \frac{wr(1 - wr)}{n - 1} = 0$$

So $w = 0$, which means that:

$$C = wb + (1 - w)wr_{t:t+tp} = 0$$

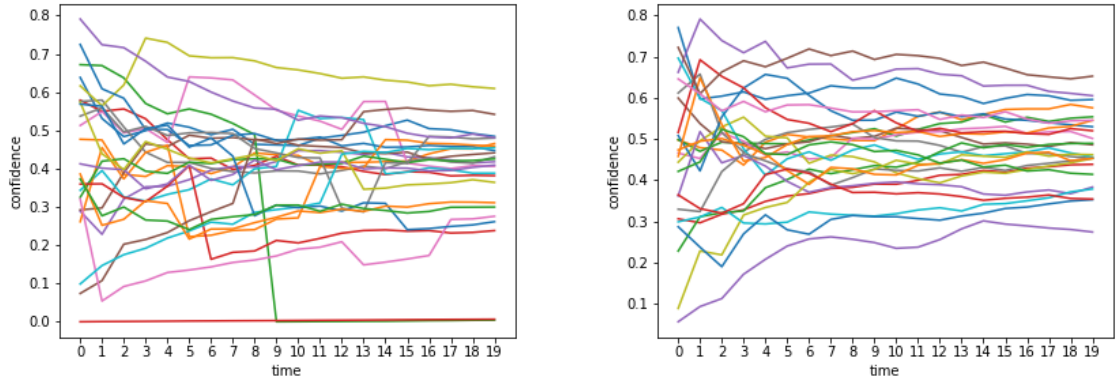
$$MSE(C) = w^2 s + (1 - w)^2 MSE(wr_{t:t+tp}) = 0$$

If we update b and s by C and $MSE(C)$, his confidence will stay at 0 and never changed. Figure 3.2.b shows there is no such issue with option 2 and everyone's confidence slowly moves to his historical win rate. Therefore in the simulation, we will leverage option 2, which keeps belief and susceptibility unchanged across time.



(a) Option 1 (b) Option 2

Figure 3.1: Every players' Confidence-win rate difference across time



(a) Option 1 (b) Option 2

Figure 3.2: Every players' Confidence across time

3.1.4 Confidence & susceptibility relation

In the last section, when running the simulation, we set confidence and susceptibility are independent of each other and all sampled from U(0,1). However, in reality, high confidence people is probably also stubborn. In other words, their susceptibility is small. On the opposite, low confidence people is probably infirm, so their susceptibility is relatively large.

To follow the reality, for susceptibility, instead of sampling from U(0,1), we set

$$susceptibility \sim |\frac{1}{N(10C, 1)}|$$

Further more, in simulation, we hypothesis that high knowledge people are more likely to be confident, and low knowledge people are more likely to be low in confidence. So given one's knowledge k, we set his initial belief b follows truncated normal distribution between [0,1]:

$$b \sim N(k, 1), b \in [0, 1]$$

By this design, a knowledgeable person may more willing to choose bet when two options have equal utility. For example, if a knowledgeable person's historical win rate is 50%. Given win = 20, loss = -10 and fix = 5, irrespective of loss aversion and risk aversion, he would very likely to enter into the bet. By credibility theory, his confidence is $wb + (1 - w) * 0.5$, which is larger than 0.5 as long as $b > 0.5$. As his initial belief $b \sim N(k, 1), b \in [0, 1]$, so it is very likely that $b > 0.5$, which indicates that he will choose bet.

3.1.5 Activity based confidence

The least square credibility confidence has a drawback. The confidence being calculated is a overall confidence, which means it is independent with the activity he just uploads. However, in reality, people's confidence also depends on the activity: confidence for the strong activity is higher and for the weak activity is lower. The problem is, how do a people define strong and weak activity? In this research, we purpose k nearest neighbor to address this problem:

k nearest neighbor

The main idea is finding k activities with the nearest score from historical activities. Then calculate the mean and mean squared error of these activities as a new estimate. As shown in figure 3.3, there are points, triangle points and x marker on the scatter plot, with points represent historical activities that lost, triangle points represent activities that won and x marker represents the current activity. The confidence is modified by the win rate of k nearest activities of his perception $(p_1, p_2, .., p_n)$ of the current activity in the activity space. In this simulation, we use Euclidean distance to as the metric to calculate distance:

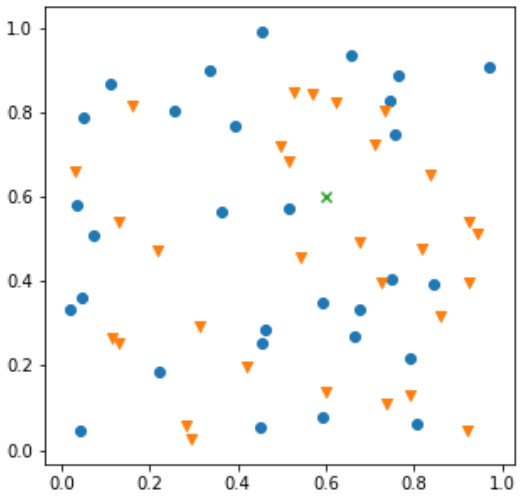
For the perception of activity 1: $P_1 = (p_{11}, p_{12}, .., p_{1n})$ and the perception of activity 2:

$P_2 = (p_{21}, p_{22}, .., p_{2n})$, the Euclidean distance between P_1 and P_2 is:

$$\sum_{i=1}^{n}(p_{1i} - p_{2i})^2$$

For example if k=6, the k nearest activities formed into a new estimate with mean= 83.3%, and squared error = $\frac{1}{36}$. We will then tune the overall confidence with new estimates using least squared credibility. Figure 3.4 shows the difference between original confidence and modified confidence, where $k = n|4$.

With this design, for players with high perceptivity, his will have high confidence with strong activities and low confidence with weak activities. For players with low perceptivity, he may by chance have a high confidence with a weak activate and low confidence with a strong activate.



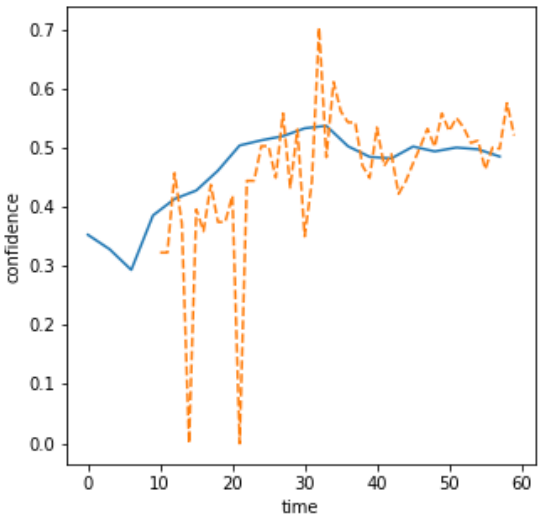


Figure 3.3: A sample of historical win & loss in activity space. Points represent historical activities that lost, triangle points represent activities that won and x marker represents the current activity

Figure 3.4: A sample of modify confidence (dashed line) and original confidence (solid line).

3.2 Prospect Theory based Decision Making Simulation

In the last chapter, every option has several dimensions of profit, the key point of the variety is that different people value these profits differently. However, in this chapter, there is only one-dimensional profit: the point. So when making decisions, instead of considering appreciation toward each dimension, people consider more about gain, loss, and probability. The key point of the variety in this chapter is that different people have different attitudes toward risk, loss, and confidence. In this section, we will present how we use prospect theory to simulate the decision-making process of whether or not entering into the bet. We will start by introducing prospect theory, then illustrate how we will use prospect theory and explain the rationale behind our approach.

3.2.1 Prospect Theory

In prospect theory, Prospect is defined as state-contingent outcomes. Usually, prospect x is denoted as $(E_1 : x_1, ..., E_n : x_n)$, yielding x_i under event E_i,..,yielding x_i under event E_i, .., and x_n under event E_n. As in our simulation, there are only two events: win and loss, so for simplicity, we use $M_p m$ to denote the prospect that has probability p to win M and probability (1-p) to loss m.

For any person, given two prospect x and y. $x \succcurlyeq y$ means this person prefer x to y; $x \sim y$ means $x \succcurlyeq y$ and $y \succcurlyeq x$. The preference is hypothesized to have transitivity, which means if $x \succcurlyeq y$ and $y \succcurlyeq z$, then $x \succcurlyeq z$. Prospects that yield the same outcome for each event are called constant prospects. In this paper, we use α to denote constant prospects and its outcome always equal to a constant number $\bar{m}$. A constant prospect α is called the certain equivalent (CE) of a prospect x if $\alpha \sim x$. In this simulation, the main problem is: if prospect x has two fixed outcomes $M > m$ and their probabilities are p and 1-p. How much should CE(x) be?

Pure rationally, the certain equivalent equal the expected value of prospect x, which is:

$$CE(x) = pM + (1 - p)m$$

However in reality, it is not. In an experiment of Tversky & Kahneman [1981] , most of people expressed the following preference:

$$240 \succcurlyeq 1000_{\frac{1}{4}}0$$
$$0_{\frac{1}{4}} - 1000 \succcurlyeq -750$$

These preference indicate that:

When $\bar{m}$, M and m > 0, even with $\bar{m} < pM + (1-p)m$, most of people prefer α than x.

In some other circumstance, When $\bar{m}$, M and m < 0, even $\bar{m} \geq pM + (1-p)m$, most of people prefer x than α.

These phenomena are called risk aversion, traditional economics explain them by introducing two concepts: utility concavity, probability weighting. Prospect theory believes there is one more component leading to these phenomena: reference dependence.

Utility concavity

Most economics researchers believe these phenomena are because of risk aversion. And they proposed to use utility function with different concavity to describe people's attitude toward risk. In other words, instead of using $CE(x) = pM + (1-p)m$, they proposed to use $U(CE(x)) = pU(M) + (1-p)U(m)$ to calculate certain equivalent.

To make readers better understand how concavity relates to risk aversion, we start by introducing how risk aversion and risk neutrality are defined. In Peter's book: Prospect theory for risk and ambiguity [2010], risk aversion, risk-seeking, and risk neutrality are defined as:

Risk aversion holds if every prospect is less preferred than its expected value. For example: $50 \succcurlyeq 100_{\frac{1}{2}}0$

Risk seeking holds if every prospect is preferred to its expected value. For example: $50 \preccurlyeq 100_{\frac{1}{2}}0$

Risk neutrality holds if every prospect is indifferent to its expected value. For example: $50 \sim 100_{\frac{1}{2}}0$

Figure 3.5 shows how risk aversion looks like.

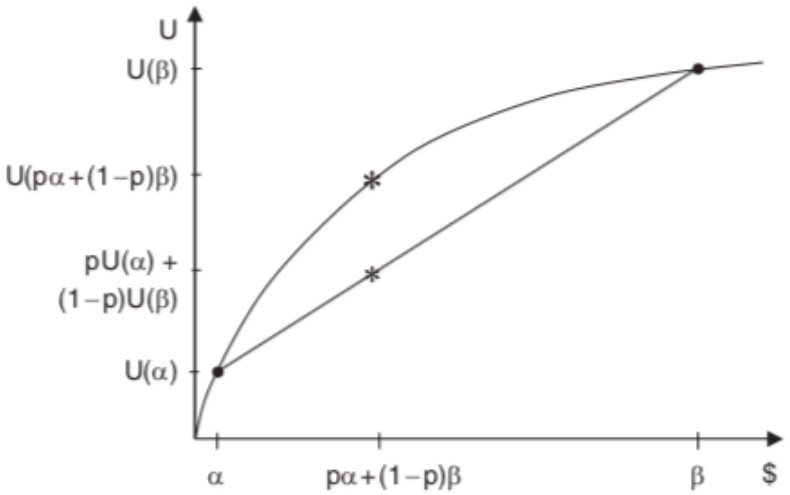


Figure 3.5: Peter [2010]: Risk aversion

We can transform these definition to the uitility form:

Risk aversion: $U(p\alpha + (1-p)\beta) \geq pU(\alpha) + (1-p)U(\beta)$, $\forall p \in [0,1]$,α,β

Risk seeking: $U(p\alpha + (1-p)\beta) \leq pU(\alpha) + (1-p)U(\beta)$, $\forall p \in [0,1]$,α,β

Risk neutrality: $U(p\alpha + (1-p)\beta) = pU(\alpha) + (1-p)U(\beta)$, $\forall p \in [0,1]$,α,β

Which are the same as the definition of the concave function, convex function, and linear function.(See Figure 3.8)

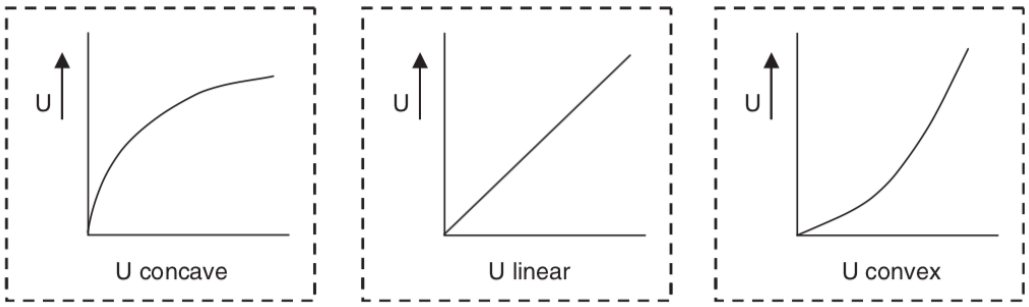


Figure 3.6: Peter [2010]: Concavity, linearity, and convexity.

So they believe:

Risk aversion $\Leftrightarrow$ U is concave

Risk seeking $\Leftrightarrow$ U is convex

Risk neutrality $\Leftrightarrow$ U is linear

Probability weighting

It is found that apart from outcome-based risk aversion, probabilistic risk aversion also exists. Kobberling & Peter [2003] examined that in some simulation, increasing risk aversion in terms of probability weighting leads to more favorable outcomes. So it is suggested that instead of using $U(CE(x)) = pU(M) + (1-p)U(m)$ to calculate certain equivalent, use the rank-dependent utility (RDU) is more appropriate. For prospect with two outcomes M¿m, rank-dependent utility is calculated as:

$$U(CE(x)) = w(p)U(M) + (1 - w(p))U(m)$$

where $w : [0,1] \to [0,1]$ is a probability weighting function that increases strictly and satisfied w(0)=0 and w(1)=1.

Reference dependence

Utility curvature and probability weighting can explain many risk aversion phenomena. however, Peter [2010] said more than half of the risk aversion empirically observed is not because of utility curvature or probability weighting. Instead, they are generated by loss aversion.

In Tversky & Kahneman[1981] experiment, people are asked to make choice from two scenarios:

$$240 \text{ versus } 1000_{\frac{1}{4}}0$$
$$0_{\frac{1}{4}} - 1000 \text{ versus -750}$$

For scenario one, the majority chose the left prospect, indicate that they are risk aversion. For scenario two, the majority chose the right prospect, which indicates that they now risk seeking.

The utility function and probability weighting function should be consistent, however, the risk attitude of people is inconsistent. Prospect theory points out that, the change of risk attitude is loss aversion. Loss aversion is the main empirical phenomenon of reference dependence, it means that total unhappiness from loss may exceed the total happiness from gain. In reference dependence, the weights of loss and gain are relative to a reference point x_0. And the reference-dependent utility function has the form: $u(x|x_0) = v(x - x_0)$, where v is risk aversion toward gain and risk-seeking toward loss. Otherwise, it will result in Rabin Paradox that if someone rejects a small positive expected value gamble, he must be extremely risk aversion and will reject all gambles no matter how much the gain is if provided with lager (for example 10 times) loss. The proof of Rabin Paradox is in the appendix. In our simulation, we the scale of the gain and loss won't change much, so it is fine we set reference point stay at 0. Given a basic utility function u and a

loss aversion parameter $\lambda > 0$, with u(0)=0, the overall Utility function U is defined as:

$$U(\alpha) = u(\alpha) \text{ for } \alpha \geq 0,\ U(\alpha) = \lambda u(\alpha) \text{ for } \alpha \leq 0$$

Loss aversion holds if $\lambda > 1$, Gain seeking holds if $\lambda < 1$. Figure 3.7 shows how the utility with loss aversion looks like.

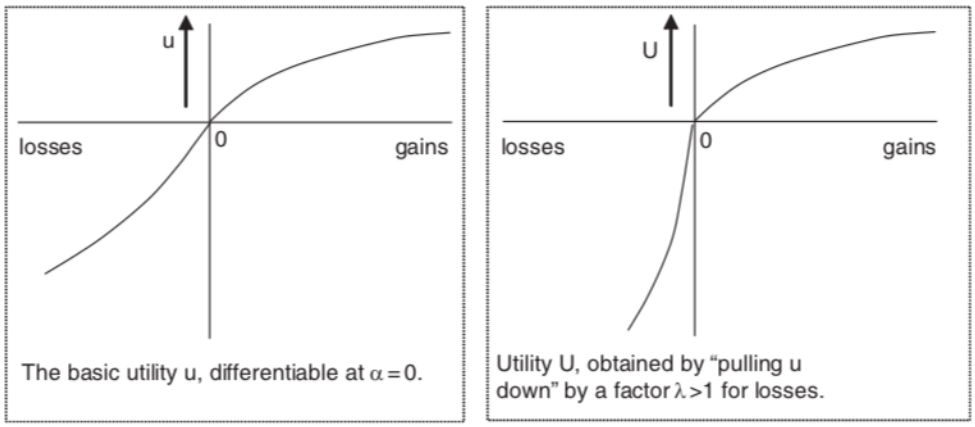


Figure 3.7: Peter [2010]: Loss aversion.

3.2.2 Utility function

There are two most popular parametric family of utility: power utility and exponential utility, which defined as following:

Power utility

For $\alpha > 0$, and given parameter denoted θ:

for $\theta > 0$, $u(\alpha) = \alpha^{\theta}$

for $\theta = 0$, $u(\alpha) = ln(\alpha)$

for $\theta < 0$, $u(\alpha) = -\alpha^{\theta}$

For $\theta > 1$, the function is convex; for $\theta = 1$, the function is linear, and for $\theta < 1$, the function is concave

Exponential utility

for $\theta > 0$, $u(\alpha) = 1 - e^{-\theta\alpha}$

for $\theta = 0$, $u(\alpha) = \alpha$

for $\theta < 0$, $u(\alpha) = e^{-\theta\alpha} - 1$

For $\theta > 0$, the function is convex; for $\theta = 0$, the function is linear, and for $\theta < 0$, the function is concave

Problems of utility functions

Consider loss aversion, the Utility is defined as:

$U(\alpha) = u(\alpha)$, if $\alpha > 0$

$U(\alpha) = -\lambda u'(\alpha)$, if $\alpha < 0$

Tversky & Kahneman [1992], Peter [2010], Kobberling & Wakker [2003] point out that for using power utility to study loss aversion, there are three significant problems:

1. Loss aversion depends on the monetary unit.
2. Not always $-U(\alpha) > U(\alpha)$ for all $\alpha > 0$

As we have only one monetary and usually the range for α that $-U(\alpha) < U(\alpha)$ is usually quite small. Using power utility is not a problem for us.

For using exponential utility, De Giogi & Hens (2006) argued that piecewise exponential utility is too shallow for extreme positive outcomes and too steep for extreme negative outcomes, which will hamper estimations of loss aversion if an extreme outcome is involved.

In our simulation, as there are no extreme outcomes involved, using exponential utility is also not a problem for us.

The proof of these problems is skipped. In this simulation, we use power utility as it is more popular compare with exponential utility and we have some empirical value found by Tversky & Kahneman [1992] and this value has been verified by lots of researches.

3.2.3 probability weighting function

A psychological phenomenon is found by lots of research (Peter[2010], Tversky,& Wakker[1995], Wu & Gonzalez[1999]) that people's probability weighting function is too shallow in the middle and too steep near endpoint. This phenomenon is called Likelihood insensitivity. In other words, if the probability of an event equal to 0.05, people's feeling of the probability is higher than 0.05, if the probability equal to 0.95, people's feeling of the probability is lower than 0.95. See Figure 3.6.

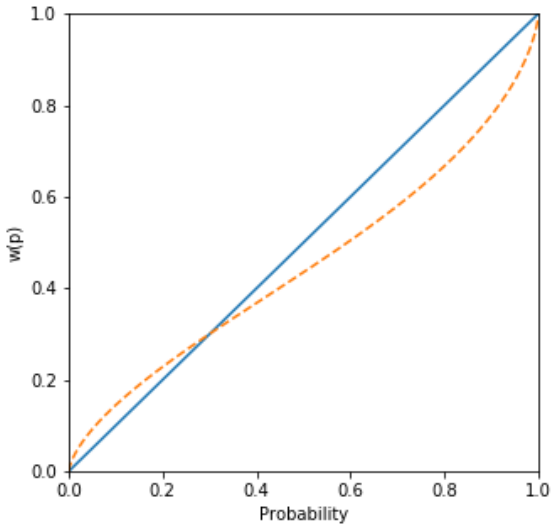


Figure 3.8: Likelihood insensitivity

There are lots of choice of the weight function to describe the inverse S shape. In this research, we considered two weighting function: Goldstein & Einhorn[1987] and Tversky & Kahneman[1992].

Goldsteim & Einhorn [1987]

$$w(p) = \frac{bp^a}{bp^a + (1-p)^a}$$

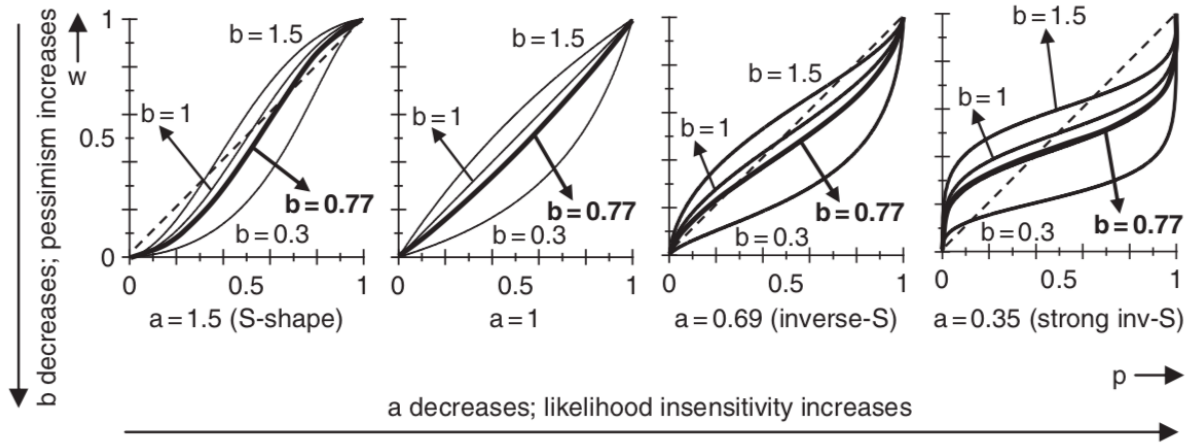


Figure 3.9: Goldstein & Einhorn [1987] family

Figure 3.9 shows the Goldstein & Einhorn [1987] family with different pairs of a and b. $a > 0$ affects likelihood insensitivity, and b is an anti-index of pessimism. $a < 1$ indicates a inverse S shape; $a > 1$ indicates a S shape; when $a = 1$, this utility function becomes a power utility; Goldsteim & Einhorn [1987] found the best parameters that fit observed data are a=0.69 and b=0.77.

Tversky & Kahneman[1992]

$$w(p) = \frac{p^c}{(p^a + (1-p)^c)^{1/c}}$$

Figure 3.10 shows the Tversky & Kahneman [1992] family with different c. for $c \geq 0.28$; for smaller c the function is not strictly increased. When c=1 the weight function is linear. Tversky & Kahneman[1992] found for gain, the best fit parameter is c=0.61 and for loss is c=0.69

In this simulation, we use Tversky & Kahneman[1992] weighting function as its empirical value is calculated with respect to loss aversion and power utility.

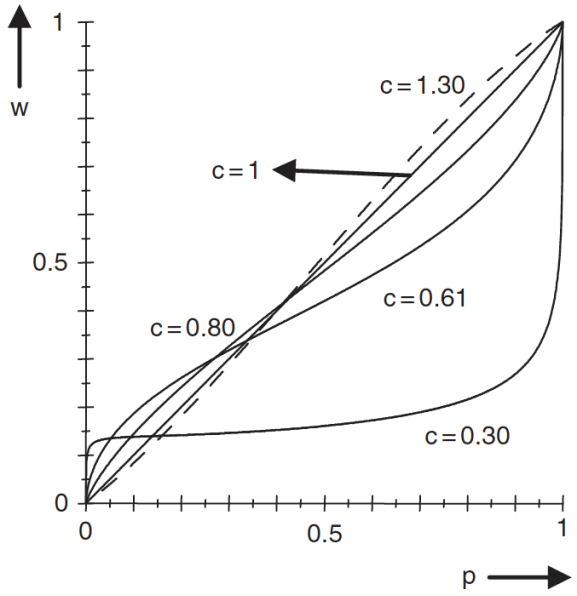


Figure 3.10: Tversky & Kahneman [1992] family

3.2.4 Parameter setting

In our simulation, every player will be randomly assigned with some parameters: loss aversion λ, risk aversion parameter toward gain θ, risk aversion parameter toward loss θ', likelihood insensitivity parameter for gain c, and likelihood insensitivity parameter for loss c' .

We hypothesis that people's risk aversion, probability weight and loss aversion are independent with all parameters defined in the last chapter. Tversky & Kahneman [1992] found the the empirical value for these parameter are: $\theta = \theta' = 0.88$, $\lambda = 2.25$, $c = 0.61$ and $c' = 0.69$. So we set:

Risk aversion parameter for gain (θ)

$$\theta = N(0.88, \sigma_\theta^2)$$

We use ρ denote the percentage of people that is risk aversion when gain. Therefore σ_θ can be calculated as:

$$\sigma_\theta = \frac{1 - \mu_\theta}{\Phi^{-1}(\rho)} = \frac{0.12}{\Phi^{-1}(\rho)}$$

where $\Phi^{-1}()$ is the quantile function of standard normal distribution.

Risk aversion parameter for loss (θ')

$$\theta' = N(0.88, \sigma_{\theta'}^2)$$
$$\sigma_{\theta'} = \frac{1 - \mu_{\theta'}}{\Phi^{-1}(\rho')} = \frac{0.12}{\Phi^{-1}(\rho')}$$

ρ' denote the percentage of people that is risk aversion when loss.

Loss aversion parameter (λ)

$$\lambda = \Gamma(k = 1.8, \theta = 1.25)$$

which calculate from mean =2.25, mode =1. Refer the emperical finding of Summer N Clay et.al [2017] Figure 3.11) and Tversky & Kahneman [1992]

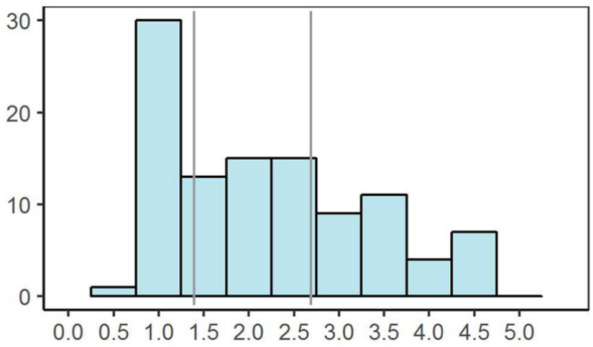


Figure 3.11: Summer N Clay et.al [2017], empirical distribution of loss aversion parameter

Parameter of Tversky & Kahneman family for gain (c)

$$c = N(0.61, 0.038), c \geq 0.28$$

Refer to the empirical finding of Adam et.al [2009]

Parameter of Tversky & Kahneman family for loss (c')

$$c = N(0.59, 0.061), c \geq 0.28$$

Refer to the empirical finding of Adam et.al [2009]

3.2.5 Decision making process

Faced with a pair of point when win G, point when loss L and point when not bet F. The player will firstly calculate his confidence c and mean squared error of c by:

$$c = wb + (1 - w)wr_{0:t}$$

$$MSE(c) = w^2 s + (1 - w)^2 MSE(wr_{0:t})$$

where

$$w = \frac{1}{1 + \frac{s}{MSE(wr_{0:t})}}$$

Next, modify c with new estimate from k nearest neighbour:

$$c' = w'c + (1 - w')wr_{knn}$$

where

$$w' = \frac{1}{1 + \frac{MSE(c)}{MSE(wr_{knn})}}$$

Finally, make decison with the option with highest utility:

$$argmax_{(G,L);F}(U(G)w(c) + U(L)(1 - w(c)), U(F))$$

where

$$U(\alpha) = \alpha^{\theta}, w(p) = \frac{p^c}{(p^a + (1-p)^c)^{1/c}}, \alpha > 0$$

$$U(\alpha) = -\lambda(-\alpha)^{\theta'}, w(p) = \frac{p^{c'}}{(p^a + (1-p)^{c'})^{1/c'}}, \alpha < 0$$

Chapter 4

Segmentation Based on Knowledge

Until now, we have illustrated all the elements that needed to run the "health food comparison". In this chapter, we present how we segment same level players into the same group. By the same level, we mean if two players lie in the same level, their winning probabilities are both approximately 50% when compared with each other. As the detailed setting of the game is in the Non-disclosure Agreement, we will only provide the necessary concept of the setting for readers to understand the simulation.

4.1 Simulation setting

In this section, we will present the detailed setting about the simulation. Firstly, we will talk about players setting, then illustrate the simulation process.

4.1.1 Players setting

The propose of this simulation is finding good indicators to segment players. We set there are in total 25 people in the game. 5 of them are high at knowledge, consistency, and perceptivity. 5 of them are low at knowledge, consistency, and perceptivity. And the other 15 players are normal people. Table 3.2 shows what we mean by high, low and normal. Their knowledge, consistency, and perceptivity will be a random number between their corresponding interval. Every player's default learning rate is a random variable sample from U(0,0.1).

The expert's weight $(w_{e1}, w_{e2}, .., w_{en})$ is set as a vector of 5 random numbers sampled from $Normal(2, 9)$. The expert is set as a person with knowledge, consistency, and perceptivity all equal to 1

Table 4.1: Simulation setting

Level	knowledge	consistency	perceptivity
High	(0.8,1)	(0.8,1)	(0.8,1)
Normal	(0.5,0.8)	(0.5,0.8)	(0.5,0.8)
Low	(0.2,0.5)	(0.2,0.5)	(0.2,0.5)

4.1.2 Simulation process

In the beginning, players will be randomly assigned to 5 different groups. In each group, every player will upload his best activity to the server. And the server will randomly generate pairs of activity from the same group to expert and players. Upon receiving activity pairs from the server, the expert and players will provide vote and reason based on the simulation model we talked about the last chapter, and the expert will also send recommendations to players. In the end, players will receive 2 pairs of vote results and reasons from experts, and 4 pairs of vote results and reasons from other players. They will adjust their appreciation based on the expert's recommendation and their own learning rate, which based on their confidence at that time point. The simulation will run 3 rounds per week, and every week we will regroup players by some rules to make every people's win rate is around 50%.

4.2 Candidate segment criteria & Segment process

In this section, we will talk about the candidate segment criteria of our segmentation rules and how we segment players based on these segment criteria.

4.2.1 Candidate segment criteria

From the prospect of the app company, there are 4 elements that can be observed and recorded from the game:

1. Players' win & loss data from the expert
2. Players' win & loss data from other players
3. Players' reason and the expert's reason toward the same activity pair
4. Players' vote and the expert's vote toward the same activity pair

These data could be engineered into several features:

1. Win rate based on expert's vote across the time
2. Win rate based on expert's vote within one month

3. Win rate based on expert's vote within one week
4. Win rate based on players' vote across the time
5. Win rate based on players' vote within one month
6. Win rate based on players' vote within one week
7. Reason similarity defined in 2.6.1
8. Reason similarity defined in 2.6.1 within one month
9. Rate of player's vote agreed expert's vote
10. Rate of player's vote agreed expert's vote within one month

We will use these feature as the segment criteria.

4.2.2 Segment process

when run the simulation, We will regroup players every week. Given a segment criteria, at the end of every week, we calculate each player's value under the segment criteria, then sort players by these values, and regroup them in order.

4.3 Segment analyze

when running the simulation, We will regroup players every week. Given a segment criterion, at the end of every week, we calculate each player's value under the segment criteria, then sort players by these values, and regroup them in order.

4.3.1 Evaluate metrics

We mainly focus on two properties of a given segment criterion: accuracy, speed, and the win rate after segmentation.

accuracy

As the game consists of 5 high-level players, 5 lower level players, and 15 normal level players. So our regroup algorithm should be able to segment all high-level players into one group and all low players into another group. The accuracy of the segment is considered as a metrics to evaluate the regroup algorithm. We use the percentage of players been successfully classified into their corresponding group as the accuracy of regrouping. The accuracy may different var time, the mean accuracy across the whole time is considered the index of overall accuracy.

speed

We use the number of weeks when for the first time all players are correctly grouped as the index of the speed.

win rate after segmentation

Our main goal is to segment players so that everyone's win rate equal to 50%. For high-level players and low-level players, we know it is the best to group them together. However, for normal level players, we should not only make sure they are in the correct group but also their win rate should all approximate to 50 %. As every run contain 20 weeks, We use the absolute deviation from 50 % of the last 10 weeks average win rate of normal level players as another metrics to evaluate the performance of the model.

4.3.2 Result and analyze

To get a stable estimation, for every criterion, we run the simulation 100 times with each run contain 20 weeks. Table 4.2 shows the segmentation result with different criteria. We can see that for classifying high-level players, reason similarity and vote agree rate are very good indicators. Both of them have fast speed and high accuracy. However, the performance of these two criteria on classifying low-level players and keeping win rate at 50% is not as good as the performance of win rate from the expert. The win rate from players has no use in classifying people as two criteria related to it all got the lowest speed and accuracy.

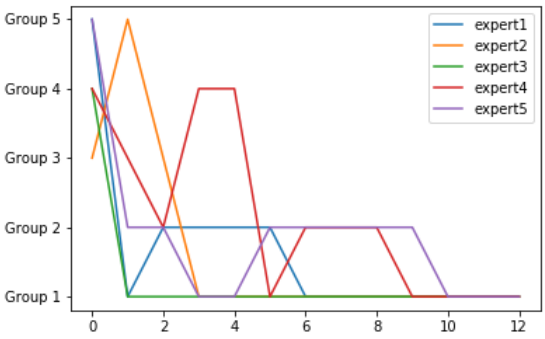


Figure 4.1: High level player movement across time

Also the stability of convergence needs to be considered. Figure 4.1 shows a sample of the movement of high level players. Usually, a stable segmentation will also have a high

accuracy across time, figure 4.2 shows the average performance every week using different criteria. It presents the convergence stability of the segmentation, the high value it gets, the more stable the segmentation is. We can see that criteria based on the whole time usually outperform the criteria based on only recent one month in the convergence stability.

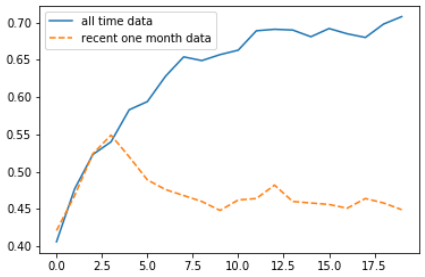
From the result, we can see that win rate from expert, reason similarity and vote agree rate are pretty good indicators. As the win rate from the expert within one month has the lowest win rate deviation, we will deeply test these four criteria in the next subsection.

Table 4.2: Segmentation result

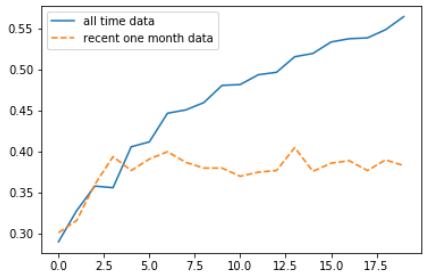
Candidate criteria	First time all success (High level)	Average accuracy (High level)	First time all success (Low level)	Average accuracy (Low level)	Win rate deviation after segmentation
Win rate from expert	11.76	0.617	9.76	0.694	4.92%
Win rate from expert, one month	18.1	0.463	14.93	0.498	4.50%
Win rate from players	19.4	0.556	16.8	0.672	5.40%
Win rate from players, one month	20	0.306	20	0.46	5.10%
Reason similarity	4.8	0.854	8.6	0.677	6.11%
Reason similarity, one month	6.07	0.759	10.5	0.603	5.77%
Vote agree rate	4.9	0.779	14.3	0.654	5.94%
Vote agree rate, one month	9.86	0.642	16.2	0.530	5.58%

4.4 Scenario test

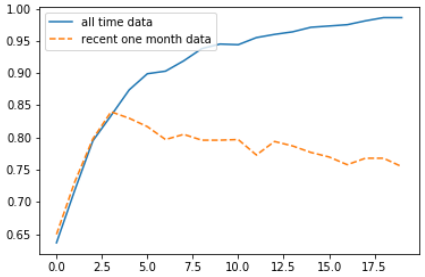
In this section, we will do more test with these four criteria.



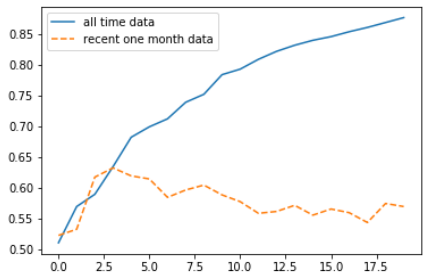
(a) Win rate based on expert's vote



(b) Win rate based on players' vote



(c) Reason similarity



(d) Vote agree rate

Figure 4.2: Average performance in different week

4.4.1 Expert is not a god

In our simulation, we set the expert knowledge, consistency, and perceptivity all equal to 1. However, in reality, even the expert is very knowledgeable, he may have some bias on the perception of foods because expect can't know every component food exactly. So we set the expert's perceptivity equal to 0.9 and scale every players' perceptivity by 0.9. The result is shown in Table 4.3. As we can see that, the accuracy of classifying high-level player of reason similarity and vote agree rate is decreased however is still the highest. Win rate from the expert is the best criteria for classifying low-level player and has the lowest win rate deviation.

4.4.2 All random

In the last several simulations, we predefined there are 5 high-level players, 5 low-level players, and 15 normal level players. What if we set every player's parameters are random

Table 4.3: Segmentation result

Candidate criteria	First time all success (High level)	Average accuracy (High level)	First time all success (Low level)	Average accuracy (Low level)	Win rate deviation after segmentation
Win rate from expert	10.5	0.64	9.23	0.659	4.97%
Win rate from expert, one month	18.1	0.469	17.44	0.49	5.09%
Reason similarity	4.8	0.82	13.03	0.64	6.60%
Vote agree rate	7.33	0.75	14.93	0.627	6.34%
Sum 1 & 3 & 4	6.47	0.838	12.43	0.737	5.80%

Table 4.4: Segmentation result

	Win rate from expert	Win rate from expert ,one month	Reason similarity	Vote agree rate
Win rate deviation after segmentation	5.61%	6.18%	6.46	6.48

numbers? In this subsection, we set every player's knowledge, consistency and perceptivity are sampled from the truncated Normal distribution with mean=0.7, variance=1, lower bound =0.2, and upper bound =1. As we introduce higher randomness to players' level, we run simulation 200 times instead of 100 to achieve a more robust result. Also, in this simulation, we still assume the expert is not a god as we believe this assumption follows the reality.

As in this simulation, we don't know the level of players, there are no metrics to evaluate the speed and accuracy. So the only metrics is the win rate deviation. Instead of only calculating the deviation of normal level people, in this subsection, we calculate everyone's deviation.

As is shown in Table 4.4, "win rate from the expert" has the lowest deviation.

4.4.3 Shuffle players every month

In this simulation, we want to test the performance of these criteria if we shuffle players into new groups at the end of each month.

We can see from Table 4.5 that, "win rate from the expert" and "one-month win rate from the expert" all have a higher speed compared with other simulations. This is because, in this simulation, high-level players have more chance to be grouped with lower level players. The ranks of different criteria in each metrics remain the same.

Table 4.5: Segmentation result

Candidate criteria	First time all success (High level)	Average accuracy (High level)	First time all success (Low level)	Average accuracy (Low level)	Win rate deviation after segmentation
Win rate from expert	8.07	0.657	7.47	0.672	5.05%
Win rate from expert, one month	8.01	0.471	6.02	0.52	5.06%
Reason similarity	5.47	0.85	8.03	0.701	6.20%
Vote agree rate	6.81	0.772	10.59	0.625	6.25%

4.5 Our approach

From all simulations we did in the previous we can get a conclusion that: reason similarity and vote agree rate are good indicators for classifying high-level players, while the win rate from the expert is good at group normal and low-level players.

So the better approach should be:

criteria = (w_1"Win rate from expert" $+w_2$Reason similarity $+w_3$Vote agree rate)/($w_1 + w_2 + w_3$)

when the player is likely to be high level, w_2 and w_3 is higher, when the player is not likely to be high level, w_1 is higher.

Table 4.3 shows the result when $w_1 = w_2 = w_3 = 1$, this new criteria got the highest accuracy on classifying both high level and low-level players. The speed is not the fastest but the second fast. Win rate deviation is on the average level.

One idea of tuning w_i is finding the threshold th_1, th_2 for reason similarity and vote agree rate classify someone is a high-level player. If the reason similarity $< th_1$, w_1 is higher, so does th_2 and w_2. We run simulations repeatly and find $th_1 = 2.33$ and 0.78.

The approach 1

For player i

$w_{1i} = 1$ if reason similarity < 2.33, else $w_{1i} = 0$
$w_{2i} = 1$ if reason similarity > 0.78, else $w_{2i} = 0$
$w_{3i} = 1$ if $w_{1i} + w2_i = 0$, else $w_{3i} = 0$

The approach 2

For player i

$w_{1i} = 2$ if reason similarity < 2.33, else $w_{1i} = 1$
$w_{2i} = 2$ if reason similarity > 0.78, else $w_{2i} = 1$
$w_{3i} = 2$ if $w_{1i} + w2_i = 2$, else $w_{3i} = 1$

However, in reality, the th_1 and th_2 will be different. If there is no data that high-level players already been labeled, it is hard to find these thresholds. We purposed another idea. Get the rank for reason similarity, the rank of reason agree rate. If one player's rank of reason similarity is small, w_1 is higher. So does w_2

The approach 3

For player i

$w_{1i} = 2$ if rank(reason similarity)≤ 5, else $w_{1i} = 1$
$w_{2i} = 2$ if rank(reason similarity)≤ 5, else $w_{2i} = 1$
$w_{3i} = 3$ if $w_{1i} + w2_i = 2$, else $w_{3i} = 2$

The approach 4

For player i

$w_{1i} = 1.5$ if rank(reason similarity)≤ 5, else $w_{1i} = 0.5$
$w_{2i} = 1.5$ if rank(reason similarity)≤ 5, else $w_{2i} = 0.5$
$w_{3i} = 2$ if $w_{1i} + w2_i = 1$, else $w_{3i} = 1$

The approach 5

For player i

$w_{1i} = 1$ if rank(reason similarity)$\leq$5, else $w_{1i} = 0$
$w_{2i} = 1$ if rank(reason similarity)$\leq$5, else $w_{2i} = 0$
$w_{3i} = 1$ if $w_{1i} + w2_i = 1$, else $w_{3i} = 0$

The approach 6

For player i
$w_{1i} = 1/(10+\text{rank(reason similarity}_i)$
$w_{2i} = 1/(10+\text{rank(rate agree rate}_i)$
$w_{3i} = 1/(10+25\text{-rank(win rate}_i)$
The result is shown in table 4.6. The approach 3 and 6 stand out with high accuracy, fast speed and low deviation.

Table 4.6: Segmentation result

Candidate criteria	First time all success (High level)	Average accuracy (High level)	First time all success (Low level)	Average accuracy (Low level)	Win rate deviation after segmentation
Sum 1 & 3 & 4	6.47	0.838	12.43	0.737	5.80%
approach 1	11.03	0.78	16.37	0.70	5.56%
approach 2	10.07	0.803	12.7	0.723	5.21%
approach 3	**7.56**	**0.81**	**12.93**	**0.740**	**5.05**%
approach 4	11.33	0.734	12.83	0.723	4.82%
approach 5	17.4	0.61	16.36	0.63	4.79%
approach 6	**9.63**	**0.823**	**12.03**	**0.747**	**5.32**%

Chapter 5

Segmentation Based on Confidence

In this chapter, we mainly focus on constructing game to identify people's confidence. It is believed that in the same level group, self-aware people is usually healthier than people who is not. So in this chapter, we present how we detect high-confidence or low-confidence players by changing the win-loss-fix point combination presented to them. We will first present how we find the best combination for a given targeted confidence, then we will illustrate our method to estimate one's confidence by comparing his rate of bet with some threshold.

5.1 Win, loss, fix combination

In this section, we mainly focus on answering the following question: Faced with a combination of win W, loss L, and fix F, the person would generate a decision based on his confidence, risk aversion parameter, likelihood insensitivity parameter, and loss aversion parameter as we introduced in 3.2.5. With risk aversion parameter, likelihood insensitivity parameter, and loss aversion parameter are all unknown, what is the best combination to approximate his confidence?

In this section, we first introduce the criteria to evaluate the efficiency and the accuracy of given combination. Then we show the result generate from simulation to illustrate how we select the optimal combination regard different level of confidence.

5.1.1 Evaluate criteria

For classification problem, directly using accuracy as evaluate criteria could be appropriate in most case. However, for imbalance data problems, employing accuracy as model criteria

is inappropriate. For example, there is such a model: all passengers took off from the airport in Canada are simply labeled as non-terrorists. It is known that Canada has an average of 100 million passengers throughout the year, and a total of 10 terrorists have been found in 2018, the model reached nearly perfect accuracy: 99.999999%. However, it is obvious that this model is not applicable as it is incapable in recognizing terrorist. To figure out this problem, we introduce four common used evaluate criteria: recall, precision,F_β score,and area under the ROC curve (AUC).

Recall

Recall was defined as:

$$Recall = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

where true and false denote whether the predictor get the same result as its true label, and positive and negative denote the predicted label. For Terrorists problem, the all-passenger-model got recall=0, which correctly indicates that the model is not appropriate at all. However, if define all passengers are terrorists, the recall become 1, which means only use recall is also not appropriate.

Precision

Precision is defined as:

$$Precision = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

So when all passengers are defined as terrorists, the recall become 1, however the precision become 0 instead.

Therefore, when evaluating a model, we need to consider recall and precision together. One very popular method is $F_\beta score$.

F_β score

F_β score is defined as:

$$F_\beta = (1 + \beta^2)\frac{precision \times recall}{\beta^2 precision + recall}$$

where β control the weight of precision and recall, which is commonly set to 1.

Area under the ROC curve (AUC)

As for classification problem, the output of model is the probability of the target belong to each class. For a binary class problem, if we change the threshold from 0 to 1 instead of fixed value 0.5, we should get a vector of different pair of true positive rate and false positive rate, where are defined as:

$$\text{true positive rate} = \textit{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

$$\text{false positive rate} = \frac{\text{false positives}}{\text{false positives} + \text{true negatives}}$$

Then a plot can be draw to describe the relationship between true positive rate and False positive rate. Figure 6.1 is a classic plot of ROC curve, the model, whose area under the ROC curve is higher than the others', generally performance better. The area under ROC curve (AUC) is a matric from 0 to 1. Intuitively, we want the area under the curse as large as it can to made the model relatively better no matter how beta in F_β score change. So the higher value the AUC is, the better the models ability in classification.

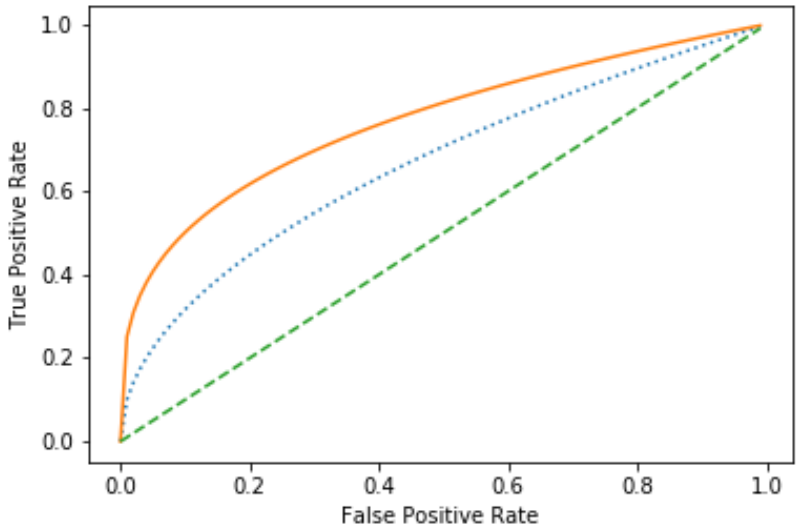


Figure 5.1: A sample of the ROC curve. Generally the dense line has better performance than densely dot line

5.1.2 Main idea

Our hypothesis is, people's decision of whether bet or not are based on his confidence, risk aversion, likelihood insensitivity, and loss aversion. So, based on the hypothesis, we can deduce that individuals who are willing to bet are more likely to be confident, and vice versa. We can also deduce that a person's knowledge won't directly affect his decisions, however it can indirectly impact the decisions as his confidence is related to his knowledge. (As we defined in 3.1.4).

So we proposed to use the rate of bet (rob), which denote the percentage of bet among all one's decisions, as the indicator of the confidence. Given a targeted confidence, there is a threshold λ compared with rob to determine whether or not someone is more confident than our targeted confidence. As the threshold is unknown, AUC is the appropriate metric for our research.

5.1.3 Simulation setting & process

As we don't know every person's risk aversion parameter, likelihood insensitivity parameter, and loss aversion parameter, so will using Monte Carlo method to find the most likely combination by running our simulation. In our simulation, we will set every one's initial brief b (defined in 3.1.2) sampled from U(0,1). As in last chapter, we have present our method to segment players into different level of groups, in this chapter, we assume our players are in the same level. Their risk aversion parameters for gain and loss, likelihood insensitivity parameter for gain and loss, loss aversion parameters, are sampled as been stated in 3.2.4. The percentage of risk aversion people ρ is set as 75% for both gain and loss.

Given a combination (W, L, F). after running the simulation, we will get every player's rate of bet rob. We also record every player's confidence $(c_1, c_2, .., c_t)$ across time, and use the mean of confidence $(c_1, c_2, .., c_t)$ as one's overall confidence. For a given targeted confidence c_t, we calculate the AUC score for each predefined combination.

Finally, the combination with highest AUC score is chosen as the best indicator to evaluate one's confidence.

5.1.4 Result

Table 6.1 shows the result:

To detect if confidence < 0.4, win=30, loss=0, fix =10 has the best performance.

To detect if confidence < 0.3, win=35 loss=0, fix =10 has the best performance.

To detect if confidence > 0.6, win=17 loss=0, fix =10 has the best performance.

To detect if confidence > 0.7, win=15 loss=0, fix =10 has the best performance.

To detect if confidence < 0.5, win=25 loss=0, fix =10 has the best performance.

Notice that when loss < 0, the combination usually has poor performance than when loss ≥ 0. That is because every person has different degree of loss aversion, which will introduce noise when identify people's confidence. For example, given win = 25, loss=-10, and fix = 5, players with high confidence may still not willing to bet as his loss aversion parameter might be quite big.

Table 5.1: Simulation result

Confidence <0.4				Confidence <0.3				Confidence >0.6			
Combinations			AUC	Combinations			AUC	Combinations			AUC
Win	Loss	Fix		Win	Loss	Fix		Win	Loss	Fix	
20	-10	5	0.857	25	-10	5	0.866	15	-10	5	0.880
25	-10	5	0.835	30	-10	5	0.885	20	-10	5	0.911
30	-10	5	0.846	35	-10	5	0.854	25	-10	5	0.843
20	0	10	0.885	25	0	10	0.934	15	0	10	0.911
25	0	10	0.942	30	0	10	0.943	16	0	10	0.955
28	0	10	0.951	33	0	10	0.949	**17**	**0**	**10**	**0.957**
30	**0**	**10**	**0.964**	**35**	**0**	**10**	**0.967**	18	0	10	0.956
32	0	10	0.946	37	0	10	0.952	25	0	10	0.932
35	0	10	0.912	40	0	10	0.922	25	0	10	0.932

Confidence >0.7				Confidence <0.5			
Combinations			AUC	Combinations			AUC
Win	Loss	Fix		Win	Loss	Fix	
10	-10	5	0.919	15	-10	5	0.872
13	-10	5	0.901	20	-10	5	0.855
17	-10	5	0.906	25	-10	5	0.873
11	0	10	0.822	17	0	10	0.932
13	0	10	0.946	20	0	10	0.940
15	**0**	**10**	**0.960**	22	0	10	0.954
17	0	10	0.949	**25**	**0**	**10**	**0.959**
19	0	10	0.939	27	0	10	0.918

5.2 Best threshold

As in the last section, we have presented how we find the optimal combination for a given targeted confidence, in this section, we present how we select the best threshold λ defined in 5.1.2.

5.2.1 Main idea

Given a targeted confidence and win-loss-fix combination, we will test the performance of different threshold λ. We use $F_{0.5}, F_1$, and F_2 score introduced in 5.1.1 as the metric to evaluate the performance of each threshold as this is the most common choice of combining recall and precision together.

5.2.2 Result

Table 5.2-5.6 show the result for each confidence level. We use F_1, F_2, and $F_{0.5}$ as the metric, with F_2 weights more on recall and $F_{0.5}$ weights more on precision. For example, When detect who has low confidence (confidence < 0.3), if we care more about the people we found all with low confidence, then $F_{0.5}$ score is the appropriate metric; if we care more about finding all low confidence people, then F_2 score is the appropriate metric; if there is no preference, F_1 is the best choice.

From Table 5.2 we can see that: for classifying confidence < 0.4, if we use combination of win $= 30$ loss $= 0$, fix $= 10$, $\lambda = 0.7$ got the highest F_1 score; $\lambda = 0.8$ got the highest F_2 score; $\lambda = 0.4$ got the highest $F_{0.5}$ score;

From Table 5.3 we can see that: for classifying confidence < 0.3, if we use combination of win $= 35$, loss $= 0$, fix $= 10$, $\lambda = 0.5$ got the highest F_1 score; $\lambda = 0.5$ got the highest F_2 score; $\lambda = 0.2$ got the highest $F_{0.5}$ score;

From Table 5.4 we can see that: for classifying confidence > 0.6, if we use combination of win $= 17$, loss $= 0$, fix $= 10$, $\lambda = 0.3$ got the highest F_1 score; $\lambda = 0.1$ got the highest F_2 score; $\lambda = 0.6$ got the highest $F_{0.5}$ score;

From Table 5.5 we can see that: for classifying confidence > 0.7, if we use combination of win $= 15$, loss $= 0$, fix $= 10$, $\lambda = 0.5$ got the highest F_1 score; $\lambda = 0.3$ got the highest F_2 score; $\lambda = 0.6$ got the highest $F_{0.5}$ score;

From Table 5.6 we can see that: for classifying confidence < 0.5, if we use combination of win $= 25$, loss $= 0$, fix $= 10$, $\lambda = 0.7$ got the highest F_1 score; $\lambda = 0.9$ got the highest F_2 score; $\lambda = 0.3$ got the highest $F_{0.5}$ score;

Table 5.2: result for confidence < 0.4

Confidence <0.4, win = 30, loss = 0, fix =10									
λ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Recall	0.536	0.621	0.695	0.721	0.840	0.848	0.893	0.929	0.950
Precision	0.960	0.942	0.932	0.930	0.854	0.846	0.833	0.753	0.70
F_1 score	0.669	0.736	0.786	0.802	0.839	0.840	**0.857**	0.827	0.804
F_2 score	0.581	0.661	0.727	0.750	0.838	0.843	0.877	**0.884**	0.884
$F_{0.5}$ score	0.805	0.842	0.863	**0.870**	0.846	0.841	0.841	0.780	0.740

Table 5.3: result for confidence < 0.3

Confidence <0.3, win = 35, loss = 0, fix =10									
λ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Recall	0.661	0.734	0.788	0.813	0.954	0.965	0.986	0.992	0.996
Precision	0.827	0.825	0.774	0.752	0.667	0.636	0.596	0.493	0.436
F_1 score	0.703	0.751	0.757	0.754	**0.770**	0.754	0.731	0.644	0.591
F_2 score	0.671	0.735	0.769	0.781	**0.864**	0.862	0.859	0.804	0.768
$F_{0.5}$ score	0.761	**0.786**	0.761	0.747	0.703	0.676	0.642	0.543	0.486

5.2.3 Example

For example, we want to find the confident (confidence > 0.6) and non-confident (confidence < 0.4) people in the high level group. Every time a person upload his activity, We will provide one of the two win-loss-fix combinations to the person, which are win = 30, loss=0, fix = 10 and win =17, loss = 0, fix =10. For simplicity, we call the first combination s_1 and the other s_2. We could calculate the rate of bet for each person of two combinations (rob_{s1} and rob_{s2}). If we use the lambda that has the highest $F_{0.5}$ score, then:

$$confidence \begin{cases} \in [0, 0.4] & rob_{s1} \leq 0.4, rob_{s2} < 0.6 \\ \in (0.4, 0.6] & rob_{s1} > 0.4, rob_{s2} < 0.6 \\ \in (0.61] & rob_{s1} > 0.4, rob_{s2} \geq 0.6 \end{cases}$$

If $rob_{s1} \leq 0.4, rob_{s2} \geq 0.6$, this player may be is a random player.

Table 5.4: result for confidence > 0.6

Confidence >0.6, win = 17, loss = 0, fix =10									
λ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Recall	0.954	0.931	0.856	0.815	0.791	0.648	0.612	0.533	0.456
Precision	0.736	0.781	0.852	0.856	0.867	0.933	0.943	0.959	1
F_1 score	0.826	0.846	**0.849**	0.829	0.821	0.756	0.734	0.711	0.694
F_2 score	**0.897**	0.894	0.852	0.819	0.801	0.686	0.654	0.612	0.589
$F_{0.5}$ score	0.769	0.805	0.849	0.843	0.846	**0.850**	0.842	0.833	0.790

Table 5.5: result for confidence > 0.7

Confidence >0.7, win = 15, loss = 0, fix =10									
λ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Recall	0.992	0.973	0.932	0.878	0.846	0.623	0.592	0.488	0.375
Precision	0.447	0.514	0.631	0.693	0.725	0.894	0.891	0.903	0.916
F_1 score	0.600	0.653	0.731	0.748	**0.758**	0.701	0.675	0.629	0.593
F_2 score	0.773	0.800	**0.828**	0.809	0.801	0.655	0.625	0.561	0.508
$F_{0.5}$ score	0.497	0.560	0.665	0.710	0.734	**0.786**	0.767	0.748	0.741

Table 5.6: result for confidence < 0.5

Confidence <0.5, win = 25, loss = 0, fix =10									
λ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Recall	0.470	0.640	0.717	0.731	0.809	0.829	0.856	0.896	0.935
Precision	0.980	0.965	0.951	0.943	0.913	0.896	0.884	0.815	0.789
F_1 score	0.713	0.762	0.812	0.818	0.852	0.856	**0.864**	0.850	0.853
F_2 score	0.619	0.683	0.751	0.763	0.825	0.839	0.858	0.876	**0.899**
$F_{0.5}$ score	0.848	0.869	**0.888**	0.887	0.886	0.878	0.874	0.828	0.813

Chapter 6

Conclusion

Here we wish to conclude and briefly review several aspects of our simulation model. We will begin in Section 5.1 with a summary of what we did and our achievement. This is followed by a discussion of potential issues of our simulation and possible improvements. Finally, in Section 5.3, we have a brief look at potential directions for future work.

6.1 Summary

In this research, we design several models to mimic the decision-making process of different types of people, aiming at simulating the process of a given game context.

For simulation models of decision-making process given multiple criteria, we use weighted sum scoring system with interaction and kernel function as the basic structure, which is rational and general to cover most of the real-life situations. Furthermore, this design of structure makes our models capable to generate reasons and recommendations based on the decision. More importantly, our models are compatible to work with each other and learning from each other. Our modular design makes it easy in the future to replace any possible module, for example, a different and appropriate scoring system.

For the decision-making process of the gamble, we leverage credibility theory to update people's general confidence and utilize k nearest neighbor algorithm to modify the confidence based on the specific object. Moreover, instead of expected utility, we employ prospect theory to simulate the phenomenon of risk aversion, likelihood insensitive, and loss aversion when individuals make the decision. Furthermore, We present a rational method to sample the general population based on the empirical study, so that the simulation result is more close to the reality.

Based on the simulation results, We produced a reasonable, fast, and accurate segmen-

tation model, so that same level people are grouped together. We did several scenario tests to study the pros and cons of different segmentation criteria. We found that reason similarity and vote agree rate are good indicators for classifying high-level players, while the win rate from the expert is good at grouping normal and low-level players. We proposed several approaches to combine these criteria together, test their performance and give two approaches that good at segment every level of people.

Finally, we present how we constructing game to identify people's confidence. We use the AUC score to find the best combination of a given targeted confidence, and find the best threshold for detecting a given confidence and combination.

6.1.1 Potential issues

One potential issue of our simulation models in chapter 2 is, these models are all based on the same score system: weighted sum scoring system with interaction and kernel function. This system already covers most of the situations, however, if people's activity score is based on the highest or lowest score in different criteria, our current system can't simulate this process. Wierzbicki [1980] introduced the achievement secularizing function to figure out this problem, which defines several functions to calculate the score of an activity and choose the lowest score as the final score. We didn't employ this method as it introduced too much freedom and make it harder to generate the general population. To figure out this issue, we proposed the decision tree based model to mimic the general population, with different level of people has a different number of training set generate form expert. However, we couldn't find an appropriate method to generate recommendations, and learn from others based on this system. Our current model is the most compatible model to cover all the necessary part of the simulation, and it intuitively follows the process for healthy activity selection. For a more general decision-making process, our model may have a little bias.

Another potential issue is currently, we use the exhaustion method to generate general population. AS is illustrate in chapter 2, we randomly generate parameters of person until they satisfied some condition. This method is feasible but lack of efficiency. For very large population setting, this method is inadvisable as the speed of simulation becomes a problem.

The third potential issue is the way we generate population. Currently, it is based on empirical research. However, In reality, the true distribution of these parameters may be different, as our simulation is targeting to a specific gaming context.

6.1.2 Future work

Fixing the three potential issues is one of our future works. Our method of using decision tree model to mimic players' behavior is a workable solution to fix the first potential issue. However, an appropriate way of generating recommendation and learning is still under construction.

Another work could be done in the future is designing a better model to find a person's confidence. In our model, we use combination and threshold to classify the range of a player's confidence. So our method is not able to generate a likelihood for our estimate and we can't give a specific number of confidence. One possible solution is using Bayesian theory. Let $s_1, s_2, .., s_n$ denote the decisions made by a player when faced with different combinations, then our main target is calculating the probability of the player being high level, normal level, and low level, which are P(high level $|s_1, s_2, .., s_n$), P(normal level $|s_1, s_2, .., s_n$), and P(low level $|s_1, s_2, .., s_n$). Using Bayesian theory we can imply that:

$$P(\text{high level}|s_1, s_2, .., s_n) = \frac{P(s_1, s_2, .., s_n|\text{high level})P(\text{high level})}{\sum_{some=low}^{high} P(s_1, s_2, .., s_n|\text{some level})P(\text{some level})}$$

P(high level), P(normal level), and P(low level) could be found by survey or experiment. Then we just need to find $P(s_1, s_2, .., s_n|$some level). However, as $P(s_1|$some level), $P(s_2|$some level),..,$P(s_n|$some level) are not independent, this is not an easy job. So we proposed to put them into future work.

The third future work is segmentation by one's learning rate. Ultimately, we hope to segment players by their knowledge, confidence and learning rate. However, as time limits, we put them as our future work. In here, we just give our ideas of estimating one's learning rate: 1. track one's reason similarity and rate agree rate to see if there is a trend of getting smaller; 2. track expert's recommendation and calculate the similarity between consecutive recommendations. If the expert always gives the same recommendation to a player, this player may not a recommendation taker.

References

[1] Wierzbicki A. *The Use of Reference Objectives in Multiobjective Optimization.* Lecture Notes in Economics and Mathematical Systems. Springer, Berlin, 177: 468486, 1980.

[2] Harris A. M. & Reed C. L. Clay S. N., Clithero J. A. *Loss Aversion Reflects Information Accumulation, Not Bias: A Drift-Diffusion Model Study.* Frontiers in psychology, 8, 1708., 2017.

[3] Saaty T Gass S. *Parametric Objective Function Part II.* Operations Research, 3: 316319, 1955.

[4] Figueira J. & Ehrgott M Greco, S. *Multiple criteria decision analysis.* Springer, New York, 2016.

[5] Zadeh L. *Fuzzy Sets.* Information and Control., 8(3): 338353, 1965.

[6] S Zionts. *MCDM: If Not a Roman Numeral, then What?* Interfaces, 9(4), 94-101, 1979.

APPENDICES

Appendix A

Matlab Code for Making a PDF Plot

A.1 Using the Graphical User Interface

Properties of Matab plots can be adjusted from the plot window via a graphical interface. Under the Desktop menu in the Figure window, select the Property Editor. You may also want to check the Plot Browser and Figure Palette for more tools. To adjust properties of the axes, look under the Edit menu and select Axes Properties.

To set the figure size and to save as PDF or other file formats, click the Export Setup button in the figure Property Editor.

A.2 From the Command Line

All figure properties can also be manipulated from the command line. Here's an example:

```
x=[0:0.1:pi];
hold on % Plot multiple traces on one figure
plot(x,sin(x))
plot(x,cos(x),'--r')
plot(x,tan(x),'.-g')
title('Some Trig Functions Over 0 to \pi') % Note LaTeX markup!
legend('{\it sin}(x)','{\it cos}(x)','{\it tan}(x)')
hold off
set(gca,'Ylim',[-3 3]) % Adjust Y limits of "current axes"
set(gcf,'Units','inches') % Set figure size units of "current figure"
set(gcf,'Position',[0,0,6,4]) % Set figure width (6 in.) and height (4 in.)
cd n:\thesis\plots % Select where to save
```

```
print -dpdf plot.pdf % Save as PDF
```