Chapter 8
Link Analysis

Link analysis has been recognized as an effective technique in data science to explore the relationships of objects. The objects can be social events, people, organization and even business transactions. This chapter reports the practical models of link analysis in various data-driven application areas. Section 8.1 presents a recommendation system for marketing optimization [1]. Section 8.2 is about advertisement clicking prediction [2]. Section 8.3 presents a model for customer churn prediction [3]. Section 8.4 provides node coupling clustering approaches for link prediction [4]. Finally, Sect. 8.5 discusses a pyramid scheme model for consumption rebate frauds [5].

8.1 Recommender System for Marketing Optimization

This section proposes a new method called trigger and triggered (TT) model. It aims to solve the problems described above and provide a lifecycle recommendation. The proposed method consists of two parts. The first part eliminates concentration noise in the training data and can be used as independent anonymous recommendation system. The second part serves as a recommendation system with lifecycle awareness among products. In the following, Sect. 8.1.1 introduces terminologies and related techniques while Sect. 8.1.2 describes the proposed model.

8.1.1 Terminologies and Related Techniques

8.1.1.1 Score Matrix

For convenience of discussion, this section uses the term "score-matrix" to describe the score between different stock keeping units (SKU, a unique identification of

Y. Shi, *Advances in Big Data Analytics*,
https://doi.org/10.1007/978-981-16-3607-3_8

each products). The score of SKU can be the number of times of co-purchases, or that of sequential purchases (a customer buys B because of the product A is bought first) or the similarity of products.

8.1.1.2 Weibull Distribution

The Weibull distribution [6] is one of the most widely used lifetime distributions. It was originally proposed by the Swedish physicist Waloddi Weibull, who used the model to approximate the distribution of breaking strength of materials. The versatility of distribution can take on the characteristics of other types of distributions, by tuning value of the shape parameter [6]. The probability density function of a Weibull model with a random variable x is shown in (8.1):

$$f(x) = \frac{\alpha}{\beta}\left(\frac{x-\mu}{\beta}\right)^{\alpha-1} e^{-\left(\frac{x-\mu}{\beta}\right)^{\alpha}} \tag{8.1}$$

where α, β and γ are known as the shape, scale and threshold parameters, respectively with constraints that $\alpha > 0$, $\beta > 0$, $\gamma > 0$. The advantage of Weibull distribution is its versatility of distribution. Therefore, in this section we use the characteristics to simulate distributions with a left long tail or right long tail, that looks like the sequential consumptions from a customer.

8.1.1.3 Gradient Descent

Gradient descent [7] is an optimization algorithm. It uses the method of first-order iterative gradient optimization to find the minimum of a function. To find a local minimum of a function, small steps are taken forward from the negative direction of the gradient of the function at current points. Stochastic gradient descent is a stochastic approximation of the gradient descent by aggregating a batch of gradient descent of sample data, which can largely increase the speed of parameter tuning [8]. Conjugate gradient descent derives from gradient descent, but instead of using gradient descent, it applies conjugate directions in the process of optimization [9].

8.1.1.4 Loss Function and Measurement

There are several metrics in the evaluation of recommender system [10, 11]: accuracy, coverage and diversity. Accuracy is the most important metric in a recommender system. A recommender system for top-k will give top k items to users in a sequence. There are multiple ways to measure its accuracy, for example:

$$precision@k = \frac{\left|T_{clicked} \cap T_{K,recommended}\right|}{K} \tag{8.2}$$

where $T_{clicked}$ is the items have been clicked in the test set for a user, $T_{K,\ recommended}$ is the k items recommended to a user.

If the rank of recommendation items is the major concern, the metric *ap @ k* will be used (see Eq. (8.3)).

$$ap@k = \sum_{n=1}^{k} \frac{P(n)}{\min(m, k)} \tag{8.3}$$

where $p(n)$ denotes the precision at the *n*th item in the item list.

For *ap @ k* metric, the same recommendation with different ranks will give different evaluations. For example, if user bought 3 items, follows recommended item #1 and #3, then *ap@10* = (1/1 + 2/3)/3 ≃ 0.56. For the same recommendation list, if user follows item #1 and #10, then *ap@10* = (1/1 + 2/10)/3 = 0.4.

The metrics are applied to evaluate the difference between estimated and actual purchase time, which are $\hat{y}_u^c$ and y_u^c. This section uses the mean absolute error (MAE), the root means square error (RMSE) and the mean absolute percentage error (MAPE) to evaluate the overall effect. The definitions are shown in (8.4), (8.5) and (8.6).

$$\text{MAE} = \frac{1}{N_u \times N_c} \sum_{c} \sum_{u} \left| y_u^c - \hat{y}_u^c \right| \tag{8.4}$$

$$\text{RMSE} = \sqrt{\frac{\sum_c \sum_u \left(y_u^c - \hat{y}_u^c \right)^2}{N_u \times N_c}} \tag{8.5}$$

$$\text{MAPE} = \frac{1}{N_u \times N_c} \sum_{1}^{N_u \times N_c} \left| \frac{y_u^c - \hat{y}_u^c}{y_u^c} \right| \tag{8.6}$$

8.1.2 Trigger and Triggered Model

The goal of recommender systems is not only to satisfy customers but also to meet the demands of marketing. From the view of marketing, the accuracy along with the history data cannot be the only measurement, the goal of marketing is the other important metric. In addition, the other contribution in this method is to effectively connect experts' knowledge with algorithm. Unlike traditional recommender systems, Trigger and Triggered (TT) model concerns more on concentration elimination, marketing optimization and lifecycle of trigger and triggered products. The workflow moves from product to personalized granularity through two independent algorithms: TT_PAR and TT_PPE. TT_PAR is responsible for the generation of meaningful trigger and triggered pairs, and TT_PPE is for the lifecycle

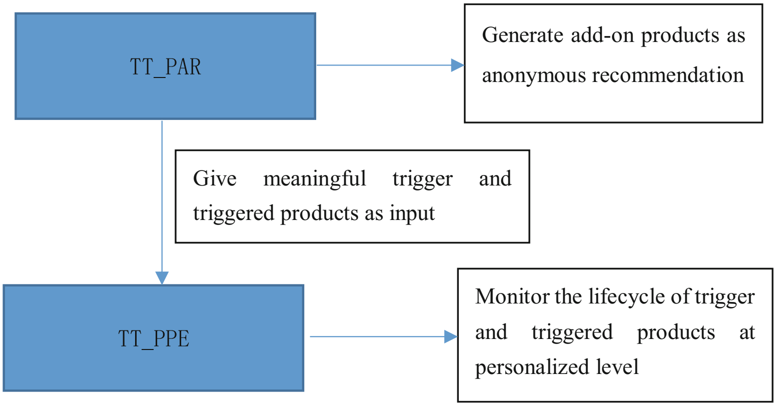

TT_PAR
Generate add-on products as anonymous recommendation
Give meaningful trigger and triggered products as input
TT_PPE
Monitor the lifecycle of trigger and triggered products at personalized level

Fig. 8.1 The workflow of TT_PAR and TT_PPE algorithms

of sequential purchases. The relationship between these two algorithms is shown in Fig. 8.1.

The algorithm TT_PAR handles eliminating concentration noise and maximizing marketing goals quite well. Trigger and triggered pairs happened in short timeframe can be treated directly as an anonymous add-on recommendation, and positive results are shown in the experiment. The TT_PPE algorithm takes trigger and triggered pairs from each customer as inputs. By combining with customers' activities and demographic information, the lifecycles of trigger and triggered products are estimated, which can be used for both onsite and offsite recommendation and promotions. The TT_PPE algorithm provides more accurate prediction on lifecycle of product pairs by comparing with other practical algorithms.

8.1.2.1 Meaningful Trigger and Triggered Pairs

When an algorithm is designed to auto recommend products, accuracy is always the most important criterion. However, focusing on accuracy alone may lead to the phenomenon of "filter bubbles", which means hot products become more popular but other options may decrease their exposure to customers as a result of sale diversity diminishing. For example, if a laptop is the most popular item in an e-commerce site and the popularity reaches to a critical point, the laptop might be treated as first recommendation option from the view of accuracy, no matter whatever the customer bought last time. This problem is even more serious when a store is featured by a certain type of products.

On the other hand, from the marketing perspective, accuracy is the primary objective. For instance, when the sales of refrigerator inexplicably decline, the recommender system should transform its role as a promotor to locate potential customers and increasing exposure of refrigerators.

Table 8.1 The example of a customer's consumption records

Date	Product id
2018-01-02	1
2018-01-02	2
2018-01-03	1
2018-01-10	3
2018-01-10	1
2018-01-10	4
2018-01-13	3
2018-01-20	3

Therefore, extracting meaningful trigger and triggered pairs in the history is important. For example, in a dataset, "iPhone–iPhone case" and "iPhone–bed" could be two pairs with the same numbers of co-purchase records. To deliver qualified results, the meaningless pair "iPhone–bed" should be excluded. This section illustrates the proposed method to run pair cleaning. The cleaned pairs can be used independently as an anonymous recommendation or combining with other information for lifecycle prediction.

8.1.2.2 Transformation of Trigger and Triggered Pairs

Trigger-Triggered (TT) pairs are product pairs of a user (at categorical level) purchased at a different time. An example of an original transaction record for user u is shown in Table 8.1.

In a store, a customer normally buys multiple products at the same day, and the same product items or new items will appear in future consumption. As mentioned above, "the same product" purchase is defined as a repetitive purchase and "new item" purchase is defined as a complementary purchase.

In general, the TT-pairs are created between purchases at a different time.

8.1.2.3 Extract Meaningful Pairs

It is necessary to design an algorithm to decrease the bias in triggered items. A TT-Paris filtering method is proposed to solve the problem. It starts with a reverted ranking approach, and applies second-order mining (a post-stage of data mining projects in which humans collectively make judgments on data mining models' performance.) [12, 13] to find meaningful pairs that are most important to the marketing. The tt-pairs filtering consists of two steps.

Algorithm 8.1 ***TT*-Pairs Filtering**

```
Step1: obtain reverted ranking using formula (8.7)
Step2: second-order mining by experts
```

Below we explain these two steps in detail.

Step 1: Rank the *tt*-pairs via *tt*-score, and extract *k* pairs with the highest grades.
The purpose of *tt*-score is to diminish the concentration bias in triggereds. *TT*-pairs are ranked by *tt*-score instead of numbers of occurrence. The *tt*-score is calculated as follows:

$$\begin{gathered} tt_{score} = nor\Big(O_{trigger_1 triggered_j}, O_{trigger_2 triggered_j}, \dots \\ , O_{trigger_n triggered_j}\Big) \\ nor\left(\cdot\right) = \frac{x - x_{min}}{x_{\max} - x_{min} + \delta} \end{gathered} \tag{8.7}$$

where O denotes occurrences and δ is a constant value used to increase weights of a list with higher occurrences.

Step 2: Experts' opinions about the best products.
To collect experts' opinions about the important products, a filtering and grading system is designed for marketing experts. The experts are required to filter the most important pairs and to grade the products.

In this system, once a category code is selected, its triggered candidates are shown in the following two sections: "Add On Products" and "Products in Selection Pool". The first section is what the expert chooses as the most important triggers and the second section includes all candidates that can be added to the selected section.

For grading system, the marketing experts will give their weights to three components: margin, quantity and price for each product (in categorical level), and observe the correctness of their grades. The final scores for products are published after their adjustment.

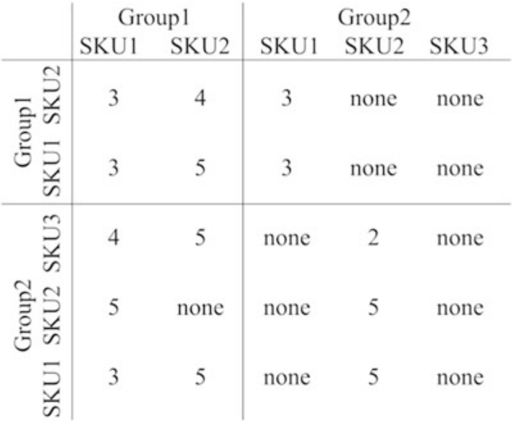
The formula of calculating score is shown in (8.8):

$$p_{score} = \alpha \times margin + \beta \times quantity + \gamma \times price \tag{8.8}$$

The product score shows the importance of products. Products with p_{score} higher than a threshold will be selected. Finally, part of the top k *tt*-pairs extracted from step 1 will be excluded if their p_{score} is less than the expert-defined threshold.

8.1.3 Trigger-Triggered Model for the Anonymous Recommendation

The anonymous recommendation should match three basic requirements: (1) reduce concentration bias in the dataset and reflect the logic correlation between products, (2) help the marketing to promote products, (3) guarantee the variety of products can be shown on the site.



		Group1		Group2		
		SKU1	SKU2	SKU1	SKU2	SKU3
Group1	SKU2	3	4	3	none	none
	SKU1	3	5	3	none	none
Group2	SKU3	4	5	none	2	none
	SKU2	5	none	none	5	none
	SKU1	3	5	none	5	none

Fig. 8.2 Scoring matrix

To realize these goals, we use both the information of the category and SKU level of products. The categorical connections between *tt*-pairs are generated based on *tt*-scores and experts' selection which is described in Sect. 8.1.1. The application of tt-scores and experts' inputs significantly reduce the problem of concentration bias. At the product level, linear programming is applied to find the right SKU pairs.

In Fig. 8.2, the matrix is the correlation score of products (tt-pairs). The size of the matrix is constrained by the generated group pair, and only relative SKUs in the groups can be connected. The scores in the matrix derive from the historical transactions. The formula of the score is shown in (8.9):

$$ppScore = CO - Occurrence + \alpha \times p_{score} \tag{8.9}$$

where $CO - Occurrence$ is the number of co-occurrence of these two products in the last 1 year. The definition of co-occurrence is that the trigger and triggered products have been bought at the same time or the triggered products have been bought within 30 days after trigger products bought firstly. The linear programming problem (shown in (8.10)) is aimed to choose the best scores among all pairs while satisfying the goals of marketing promotion and recommendation variety will be reached.

$$\begin{aligned} &\text{maximize} \quad Z = ppScore_{ij} \times \beta_{ij} \\ &\text{subject to} \begin{cases} \beta_{ij} = 0 \text{ or } 1 \\ \sum\limits_{i=1}^{n} \beta_{ij}^{c} \le \varphi_j \\ \sum\limits_{j=1}^{n} \beta_{ij}^{c} \le \omega_i \\ \sum\limits_{js\ is\ marketing\ promtion} \beta_{ij}^{c} = \delta_j \end{cases} \end{aligned} \tag{8.10}$$

where φ and ω are both a constant non-negative integer value, and δ is a constant positive integer value. The parameter φ is used to constrain the maximum numbers of products in each triggered category, that improves the variety of recommendation. The parameter ω is used to constrain the maximum numbers of a product can be shown as a triggered recommendation in the recommendation list. The value restricts over-recommendation on popular items. The parameter δ is used to promote specific products, makes sure the products can be shown more frequent at recommendation list.

8.1.4 Trigger-Triggered Model for Product Promotion

Managing the lifecycle of consumption is a key to the success of customer engagement. Appropriate advertising should be sent to customers at the appropriate time in order to stimulate customers' potential consumption. For example, a good promotion plan should send customers a new iPhone promotion 1 or 2 year after the consumption of an iPhone, or a case promotion should be sent much earlier once the customer bought a phone. A product trigger and triggered system has been designed to track lifecycle of products at individual level, which can be used for product promotion by real time recommendation system or digital advertising through email or ad platforms. The principle of the system design is that once a certain product has been purchased, then the tracking system is activated, and related products will be sent to customers at right time if the products have not been purchased yet at that time.

To study the time frame between two sequential purchase activities in personalized level, it is to estimate the probability of a user's consumption at a time interval $(j, j + \Delta t)$. That is the conditional probability p(T $\in (j, j + \Delta t|product\_u\_t)$): if the consumer u is going to buy product i in the future, what is the most likely time the consumer will buy. By examining the dataset, it is shown that the probability distribution of p(T $\in (j, j + \Delta t|product\_u\_t)$) is usually a long tail with left or right peak. Therefore, a normal distribution may not be a good candidate to simulate it. Alternatively, a Weibull distribution has been applied. Wang and Zhang [14] have applied Weibull distribution to predict the time frame in ecommerce application. Different from that work, in the section we use a Weibull model with three parameters to predict the time frame. Threshold parameter is included to deal with the case that some triggered items normally are not purchased immediately after the consumption of trigger products. In addition, we use gradient descent approach to tune the parameters instead of variational inference proposed in [14]. Even though gradient descent inference takes more time to locate a minimum, it will be easier to derive the algorithm. The probability density function of a Weibull model with random variable x is shown in (8.11).

As indicated in the formula, the scale parameter β is transformed to be a linear function of variables $\beta^T X$, where X is a vector of variables to capture signals of purchase, including a binary value indicating if the customer bought any same product or similar products in time bins t_1, $t_2 \ldots t_m$, or if triggereds have been purchased during promotion dates, or seasonality information, and etc. To make sure the derived scale parameter $\beta_1^T X > 0$, let's transform $\beta_1^T X$ to be $e^{\beta_1^T X}$. The derived density equation is:

$$f(y) = \frac{\alpha}{e^{\beta_1^T x}} \left(\frac{y-\mu}{e^{\beta_1^T x}} \right)^{\alpha-1} \exp\left(-\left(\frac{y-\mu}{e^{\beta_1^T x}} \right)^{\alpha} \right) \tag{8.11}$$

For each *i*th observation at each specific *tt*-pair c, the density function of purchase time is shown in (8.12):

$$p\left(y_i^c | X_i^c, \alpha^c, \beta^c, \mu^c\right) = \frac{\alpha^c}{e^{\beta_1^{cT} x^c}} \left(\frac{y_i^c - \mu^c}{e^{\beta_1^{cT} x^c}} \right)^{\alpha^c - 1} \exp\left(-\left(\frac{y_i^c - \mu^c}{e^{\beta_1^{cT} x^c}} \right)^{\alpha^c} \right) \tag{8.12}$$

The distributions of parameters are: $\alpha^c \sim N(\mu_\alpha, \delta_\alpha)$, $\mu^c \sim N(\mu_\mu, \delta_\mu)$, $\beta^c \sim N(\mu_\beta, \delta_\beta)$. It is denoted that $\omega = (\mu_a, \delta_a, \mu_\mu, \delta_\mu, \mu_\beta, \sum_\beta)$.

To solve the equation, we build a model separately for each product group m, where group refers to products at a particular categorical level. Therefore, in each group, the parameters are $\varphi = (\{\alpha^1, \mu^1, \beta^1\}, \{\alpha^2, \mu^2, \beta^2\}, \ldots, \{\alpha^m, \mu^m, \beta^m\})$. By grouping pairs, the purchase signal of similar products in a group is used. The joint likelihood for all variables is extended, as shown in (8.13):

$$L\left(\varphi | D_g\right) \propto L\left(D_g, \varphi\right) = p(\omega) \prod_{c=1}^{m} p\left(\alpha^c, \mu^c, \beta^c | \omega\right) \prod_{i=1}^{n_m} p\Big(y_i^c \mid \alpha^c, \mu^c, \beta^c, X_i^c \Big) \tag{8.13}$$

Since the model contains many variables, the traditional method is computationally too expensive to get an answer. Instead, functions of parameters are replaced as constant c_i and MLE is used to estimate parameters, as shown in (8.14):

$$\begin{aligned} \varphi = \left(\left\{ \hat{\alpha^1}, \hat{\mu^1}, \hat{\beta^1} \right\}, \left\{ \hat{\alpha^2}, \hat{\mu^2}, \hat{\beta^2} \right\}, \ldots, \left\{ \hat{\alpha^m}, \hat{\mu^m}, \hat{\beta^m} \right\} \right) \\ = \operatorname{argmax}\{L\,(\text{Data}, \varphi)\} \\ = \operatorname{argmin}\left\{ \sum_{c=1}^{m} \left(c_1 \alpha^{c2} + c_2 \mu^{c2} + c_3 \beta^{c2} \right) + \right. \\ \left. \sum_{c=1}^{m} \sum_{i=1}^{n_m} -\log\left(p\left(y_i^c | X_i^c, \alpha^c, \beta^c, \mu^c \right) \right) \right\} \end{aligned} \tag{8.14}$$

The pseudocode to solve the previous equation is shown in the following Algorithm 8.2. The parameters $(\mu_a, \mu_\mu, \mu_\beta)$ are initialized in the beginning. It is hidden parameters for grouping. Then Step 1 in the algorithm is to get a local minimal value of parameters φ, that is $(\{\alpha^1, \mu^1, \beta^1\}, \{\alpha^2, \mu^2, \beta^2\}, \ldots, \{\alpha^m, \mu^m, \beta^m\})$. On the basis of φ, parameters $(\mu_a, \mu_\mu, \mu_\beta)$ are updated. These two iteration steps continue until converge.

Algorithm 8.2 Trigger-Triggered Model

```
Initialize $\left(\mu_a, \mu_\mu, \mu_\beta\right) = \left(\mu_\alpha^0, \mu_\mu^0, \mu_\beta^0\right)$
    $i = 0$
    repeat
    for $tt$-pair $c = (1, 2, \ldots, m)$ in group:
            updating parameters $(\alpha^m, \mu^m, \beta^m)$ in (8.15) on the basis of known $\left(\mu_\alpha^i, \mu_\mu^i, \mu_\beta^i\right)$
    end for
    $i = i + 1$
    updating parameters $\left(\mu_\alpha^i, \mu_\mu^i, \mu_\beta^i\right)$ in (8.16) based on known $(\alpha^m, \mu^m, \beta^m)$
    end repeat (convergence)
```

$$
\begin{aligned}
(\alpha^m, \mu^m, \beta^m) = argmin\Big\{&c_1\alpha^c - \mu_a{}^2 + c_2\mu^c - \mu_\mu{}^2 \\
&+ c_3\beta^c - \mu_\beta{}^2 + \sum_{i=1}^{n_m} - \log\left(p\left(y_i^c | X_i^c, \alpha^c, \beta^c, \mu^c\right)\right)\Big\}
\end{aligned}
\tag{8.15}
$$

$$
\begin{aligned}
\left(\mu_a, \mu_\mu, \mu_\beta\right) = argmin\Big\{&c_1\alpha^c - \mu_a{}^2 + c_2\mu^c - \mu_\mu{}^2 \\
&+ c_3\beta^c - \mu_\beta{}^2 + c_4\mu_a{}^2 + c_5\mu_\mu{}^2 \\
&+ \mu_\beta{}^2 \sum_{i=1}^{n_m} - \log\left(p\left(y_i^c | X_i^c, \alpha^c, \beta^c, \mu^c\right)\right)\Big\}
\end{aligned}
\tag{8.16}
$$

Formulas (8.15) and (8.16) are hierarchical expression of (8.14) where prior knowledge are applied, two steps of inference improve the stability of prediction.

The updating methods at here can be algorithms such as Conjugate Gradient Descent, Broyden-Fletcher-Goldfarb-Shanno or others [7]. The optimum function [15] in R has been applied for parameters inference.

The experimental study of using a transaction dataset is collected from a retail store can be found in [1].

8.2 Advertisement Clicking Prediction by Using Multiple Criteria Mathematical Programming

This section proposes a multi-criteria linear regression (MCLR) [16] and kernel-based multiple criteria regression (KMCR) [17, 18] algorithms to predict CTR of ads in a web search engine given its logs in the past.

8.2.1 Research Background of Behavioral Targeting

8.2.1.1 Concept of Click-Through Rate

This section provides the concept of click-through rate (CTR). For example: As Fig. 8.3 showing, when one user views a known Chinese website 163.com, which has an ad slot that can display an advertisement. For the media 163.com, the problem is: Which advertiser should be chosen for this ad slot? The answer can be: Choose the one with max revenue. The definition is:

$$\text{Max}\ revenue = \text{Max}\left\{CPC_i \times P\_CTR_i\right\} \tag{8.17}$$

where i represents the ith of the advertiser; CPC is the Cost per Click [The money paid to the media when one ad is clicked, set by advertiser]; and $P\ \_\ CTR$ is the Prediction CTR [Expected CTR, is given by prediction model, the predict target in this section].

Note that the CTR model is very important for the platform to keep their revenue maximum. Some machine learning based regression algorithms such as logistic regression [19], maximum entropy [20], support vector regression (SVR) [21] and conditional random field (CRF) [22] have been adopted to predict the clicks of advertisements presented for a query.

In this section, the proposed multi-criteria linear regression (MCLR) and kernel-based multiple criteria regression (KMCR) algorithms will be used for CTR prediction. Note that the regression models for CTR problems are different from classification models because the former do not need the testing process for verification while the later do. However, the clicking events prediction needs classification models as introduced below.

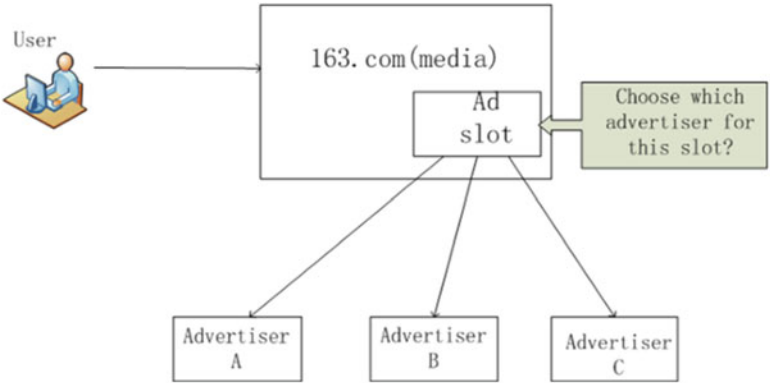


Fig. 8.3 Application scenarios of CTR

8.2.1.2 Concept of Clicking Events Prediction

For the advertiser with several candidate advertisements has the chance to display their advertisements on the ad slot, he needs to decide which ad to display. Under this scenario, a clicking events prediction model is needed to solve this problem. Through model prediction, the advertiser can learn about which ad will be clicked or not, then he can choose displaying ad that will be clicked for a good revenue. Figure 8.4 shows the application scenario of Clicking Events Prediction.

8.2.2 Feature Creation and Selection

To show the practical ability of the proposed method, the datasets of track2 of the KDD Cup 2012 are used for testing (http://www.kddcup2012.org/). The training set contains 155,750,158 instances that are derived from log message of search sessions, where a search session refers to an interaction between a user and the search engine. During each session, the user can be impressed with multiple ads, then, the same ads under the same setting (such as position, depth) from multiple sessions are aggregated to make an instance in the datasets. Each instance can be viewed as a vector (#click, #impression, DisplayURL, AdID, AdvertiserID, Depth, Position, QueryID, KeywordID, TitleID, DescriptionID, UserID). It means that under a specific setting, the user (UserID) has been impressed with the ad (AdID) for #impression times, and has clicked #click times of those. In addition to the instances,

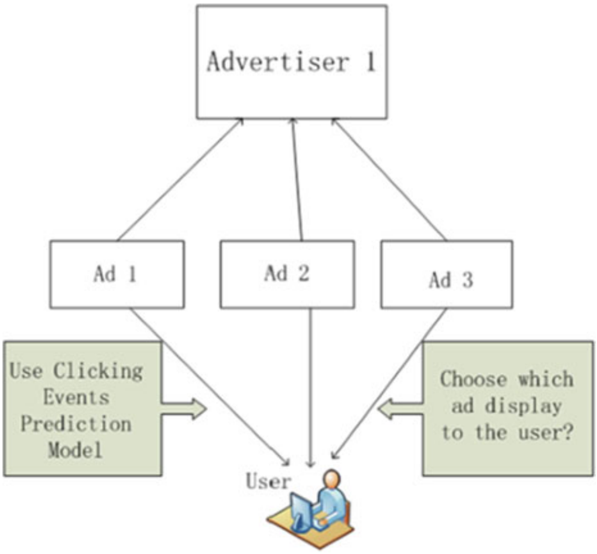

Advertiser 1
Ad 1
Ad 2
Ad 3
Use Clicking Events Prediction Model
Choose which ad display to the user?
User

Fig. 8.4 Application scenario of Clicking Events Prediction

the datasets also contain token lists of query, keyword, title and description, where a token is a word represented by its hash value. The gender and segmented age information of each user are also provided in the dataset. The test set contains 20,297,594 instances and shares the same format as the training set, except for the lack of #click and #impression. The test set is generated with log messages that come from sessions latter than those of the training set. More detailed information about the datasets can be found in (http://www.kddcup2012.org/).

However, a major challenge is to create efficient features. Feature creation and selection are the most important steps in solving a supervised learning problem. After comparing different methods, this section chooses two of them to create the features, which are called T-Set-1 and T-Set-2, respectively:

8.2.2.1 Feature Creation Method for T-Set-1

In T-Set-1, the bag of words model was used. This method is frequency-based method that is used to predict the probability of each presented word on a clicked instance based on each feature (tokens). Then, we built the whole feature space by combining the query dictionary and ad dictionary.

8.2.2.2 Feature Creation Method for T-Set-2

Two kinds of features, original feature and synthetic feature, are used for modeling in this section:

1. Original Features: The original feature set contains discrete features and continuous features. The discrete features are the unique ID of each ad, advertiser, query, keyword, tile, description, token, gender and age for one user, depth and position of ads, and the displayed URL. The continuous features are the click-through rates of each value of the discrete features. When a discrete feature is being used; the corresponding click-through rate will be activated and adopted as a continuous feature.
2. Synthetic Feature: First of all, we join any two original discrete features with each other and use them as synthetic features. We also test some 3-tuple features but only the QueryID_AdID_UserID is available. Since most 3-tuple features are too sparse and seldom activated. Secondly, we join the original discrete features with each of the tokens. Position information is added to the original discrete features to generate one 2-tuple position-based feature. Bigram features are also adopted for analyzing the queries, titles and descriptions.

8.2.2.3 Normalization

Since the ranges of all the variables' value are significantly different, a linear scaling transformation needs to be performed for each variable. The transformation expresses as:

$$x_n = \frac{x_i - \min(x_1, K, x_n)}{\max(x_1, K, x_n) - \min(x_1, K, x_n)} \tag{8.18}$$

where x_n is the normalized value and x_i is the instance value.

8.2.2.4 Categorization Method for Positive/Negative Samples

To analyze the dataset for predicting accurately, let's consider:

Advertisement 1: The time of display is 10, the time of click is 0.
Advertisement 2: The time of display is 10, the time of click is 1.
Advertisement 3: The time of display is 10, the time of click is 8.

From above, it can be seen that the gap between advertisements 2 and 3 is bigger than the gap between advertisements 1 and 2. If we simply categorize those samples based on click times, those with click times greater than 1 are categorized as positive samples and those less than 1 as negative samples, then the advertisement 2 and 3 are both labeled as 1 while the label of advertisement 1 is -1. In this situation, the influence of advertisement 2 and 3 are the same. However, as the time of click 0 and 1 is closer than 1 and 8, it is not reasonable. Therefore, we treat the click-through-rate as a probability problem. For one wonderful advertisement, someone will click it while others won't. Therefore, in this section, we calculate the click-through-rate (CTR) of each instance, and the average CTR. Then we compare each instance's CTR with the average CTR. If it is greater than the average CTR, the label of the instance should be 1; otherwise, it should be 0. The formula to calculate the CTR is described as below:

$$\begin{aligned} Click - Through - Rate(CTR) = (\#click + \alpha * \beta)\,/ \\ (\#impression + \beta) \end{aligned} \tag{8.19}$$

where $\alpha = 0.05$, $\beta = 75$, that obtained from the experiment.

8.2.2.5 Confusion Matrix

The confusion matrix is used for the performance analysis:

TP (True Positive)	the number of records in the first class that has been classified correctly
FP (False Positive)	the number of records in the second class that has been classified into the first class
TN (True Negative)	the number of records in the second class that has been classified correctly
FN (False Negative)	the number of records in the first class that has been classified into the second class

Then four different performance measures are:

$$\begin{aligned} Specificity &= \frac{TN}{TN+FP};\\ Sensitivity &= \frac{TP}{TP+FN};\\ False\ Positive\ Rate &= \frac{FP}{TN+FP};\\ False\ Negative\ Rate &= \frac{TN}{FN+TN}. \end{aligned} \tag{8.20}$$

8.2.2.6 Receiver Operating Characteristics (ROC) Graph

Receiver Operating Characteristics (ROC) graph is a useful technique for organizing classifiers and visualizing their performance. ROC graphs are commonly used in decision making, and in recent years have been increasingly adopted in the machine learning and data mining research communities. In addition to a generally useful performance graphing method, they have properties that make them especially useful for domains with skewed class distribution and unequal classification error costs. These characteristics have become increasingly important as research continues into the areas of cost-sensitive learning and learning in the presence of unbalanced classes. The reader can find details of experimental and comparison studies of the MCLR and KMCR regression models as well as the classifications MCLP and KMCP for the clicking events prediction in [2].

8.3 Customer Churn Prediction Based on Feature Clustering and Nonparallel Support Vector Machine

Bank customer churn prediction is one of the key businesses for modern commercial banks. Recently, various methods have been investigated to identify the customers who would leave away. This section presents a framework based on feature clustering and classification technique to help commercial banks make an effective decision on customer churn problem.

8.3.1 Related Work

8.3.1.1 Maximal Information Coefficient

Relationship coefficient is usually used for measuring the similarity of two variables. Person coefficient is one of the most famous relationship metrics, because it is easy to calculate and has a naive explanation. However, only linear relationship can be captured well using this metric when other kinds of dependence work badly such as sin or cubic function. Recently, [23] proposed a novel relationship measure called MIC. Inspired by innovative idea, they show that MIC could capture a wide range of associations both functional and not. Furthermore, the MIC value is roughly equal to the coefficient of determination R^2 in statistics [23]. Now we provide a little introduction to MIC.

Given a finite set D whose elements are two dimensions data points, one dimension is x-values and the other is y-values. Suppose x-values is divided into x bins and y-values into y bins, this type of partition is called *x-by-y* grid G. Let $D|_G$ represent the distribution of D divided by a *x-by-y* grid G. $I_*\ (D,x,y) = max\, I(D|_G)$, where $I(D|_G)$ is the mutual information of $D|_G$. There are infinite amount of *x-by-y* grids, so there are infinite number of $I(D|_G)$ either. Set the maximal $I(D|_G)$ as $I_*\ (D,x,y)$. Given different x- and y-value, a matrix named *characteristic matrix* could be constructed as $M(D)_{x,y} = \frac{I*(D,x,y)}{\log\ \min\{x,y\}}$. Furthermore, MIC can be obtained by $MIC(D) = max_{xy<B(n)}\{M(D)_{x,\ y}\}$, where $B(n)$ is the upper bound of the grid size. The elements in *characteristic matrix* $I_*\ (D,x,y)$ is chosen from a infinite amount of $I(D|_G)$, thus Reshef et al. develop an approximation algorithm and program for generating characteristic matrix and the estimators derived from MIC [24, 25]. With these state-of-the-art utilities, data exploration could be easily completed before other data mining procedure.

8.3.1.2 Affinity Propagation Clustering

Clustering data through similarity is a popular step in many scientific analysis and application systems. Frey and Dueck [26] developed a modern clustering method named "affinity propagation" (AP) which constructs clusters by information messages exchanged between data points. Given the similarities of each two distinct data points as input, AP algorithm considers all the instance as potential centroids at the beginning of the algorithm. And then, algorithm merges small cluster into bigger ones step by step. Different from classical clustering algorithms like k-means, each instance is regarded as one node in a network by AP clustering approach. Messages was transmitted between nodes, so each data point reconsidered their situation through new information and properly modified the cluster they belong to. This procedure went on until a good set of clusters and centroids produced.

In this process, there are mainly two categories of message exchanged between data points. One of them is sent from point i to point j which formulated as $r(i,j)$.

It illustrates the strength point i choosing point j as its centroid. The other sort information is from point j to point i as $a(i,j)$. It shows the confidence that one point j recommends itself as the centroid of another point i. And the author of AP take $r(i,j) \leftarrow s(i,j) - max_{j'\,s.\,t.\,j'\,\neq j}\{a(i,j') + s(i,j')\}$ and $a\,(i,\,j) \leftarrow \min\{0, r\,(k,k)\} + \sum_{i's.t.i'\neq i,j} \max\left\{0, r\left(i', k\right)\right\}\}$ to update current situation. Update is needed only for the pairs of points whose similarities are already known. This trait makes the algorithm much faster than other methods. To identify the centroid of point i, point j that maximizes $r(i,j) + a(i,j)$ should be considered during each iteration. AP clustering method requires a similarity matrix s as input, and the element of the matrix $s(i,j)$ provides the distance from point i to point j. In addition, the diagonal values of the matrix is not assigned 1 as usual. These values are called "preference" which show how point i is likely to be chosen as a centroid. That is to say, the larger $s(i,i)$ is, the more probability that point i play a role of a centroid. Obviously, $s(i,i)$ are key parameters which control the number of final clusters by AP method.

8.3.1.3 Nonparallel Support Vector Machine

Optimization has a long history for discovering valuable patterns and making decisions [27]. A lot of approaches have been investigated based on optimization techniques such as SVM [28], multiple criteria linear programming (MCLP) [29], etc. SVM is a serial of modern methods for data mining and pattern recognition including classification, regression and other data analytical task. Based on the theory of statistical learning, SVM use a single hyperplane to construct discriminative model which follow the Structure Risk Minimization (SRM) principle [28]. Recently, inspired by twin SVM [30, 31] (see Chap. 3 as well) proposed a novel classification method called NPSVM. Similar to twin SVM, this method applied two nonparallel hyperplanes to handle the classification problem [31].

$$f_+(x) = \omega_+^T \cdot x + b_+ = 0 \quad and \quad f_-(x) = \omega_-^T \cdot x + b_- = 0 \tag{8.21}$$

Given a binary classification dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_\ell, y_\ell)\}$ with ℓ instances and n attributes, let $\ell_+ + \ell_- = \ell$, ℓ_+ positive and ℓ_- negative instances. In order to express by matrix, the positive instances were represented by matrix $A \in \Re_{\ell_+ \times n}$. Each row of matrix A is one instance. The negatives were expressed by matrix $B \in \Re_{\ell_- \times n}$. So the primal optimization problems for NPSVM are

$$\begin{aligned} \min_{\omega_+, b_+, \eta_+, \xi_-} \; & \tfrac{1}{2}\eta_+^T\eta_+ + c_1 e_-^T \xi_- + \tfrac{1}{2}c_2\left(\|\,\omega_+\|_2^2 + b_+^2\right) \\ s.t. & -(B\omega_+ + b_+ e_-) + \xi_{\geq e_-} \geq e_-, \\ & A\omega_+ + b_+ e_+ = \eta_+, \\ & \xi_- \geq 0 \end{aligned} \tag{8.22}$$

and

$$\begin{aligned}\min_{\omega_-,b_-,\eta_-,\xi_\mp} \quad & \tfrac{1}{2}\eta_-^T\eta_- + c_3 e_+^T \xi_\mp + \tfrac{1}{2}c_4\left(\| \omega_- \|_2^2 + b_-^2\right) \\ s.t. \quad & A\omega_- + b_- e_+ + \xi_+ = e_+, \\ & B\omega_- + b_- e_- = \eta_-, \\ & \xi_+ \geq 0 \end{aligned} \tag{8.23}$$

where c_1, c_2 are the model parameters, and e_+ and e_- are the vector of ones with proper dimensions. For each hyperplane in 1, NPSVM try to make the instances of one category close to this hyperplane, and the distance between the instance of the other class and the hyperplane is more than 1 at least. The Wolf Dual problem for Eqs. (8.22) and (8.23) could be expressed as

$$\begin{aligned}&\min_{\alpha_1,\alpha_3} \tfrac{1}{2}\left(\alpha_1^T,\alpha_3^T\right) Q\left(\alpha_1^T,\alpha_3^T\right)^T - c_2 e^T \alpha_1 \\ &s.t.\ \ 0 \leq \alpha_1 \leq c_1 e,\end{aligned} \tag{8.24}$$

where

$$Q = \begin{bmatrix} BB^T & BA^T \\ AB^T & AA^T + c_2 I \end{bmatrix} + E \tag{8.25}$$

and

$$\begin{aligned}&\min_{\alpha_1',\alpha_3'} \tfrac{1}{2}\left(\alpha_1'^T,\alpha_3'^T\right) Q\left(\alpha_1'^T,\alpha_3'^T\right)^T - c_4 e^T \alpha' \\ &s.t.\ \ 0 \leq \alpha_3' \leq c_3 e,\end{aligned} \tag{8.26}$$

where

$$Q = \begin{bmatrix} AA^T & -AB^T \\ -BA^T & BB^T + c_4 I \end{bmatrix} + \begin{bmatrix} E & -E \\ -E & E \end{bmatrix} \tag{8.27}$$

The final decision could be made by comparing the distance to these two hyperplanes, respectively. The distance could be obtained from

$$\begin{aligned} f_+(x) &= \omega_+ \cdot x + b_+ \\ &= -\tfrac{1}{c_2}\left(B^T\alpha_1 + A^T\alpha_3\right)\cdot x \\ &\quad - \tfrac{1}{c_2}\left(e^T\alpha_1 + e_+^T\alpha_3\right) \end{aligned} \tag{8.28}$$

and

$$\begin{aligned} f_{-}(x) &= \omega_{-}\cdot x + b_{-} \\ &= -\tfrac{1}{c_4}\left(-A^T\alpha_1' + B^T\alpha_3'\right)\cdot x \\ &\quad -\tfrac{1}{c_4}\left(-e_{-}^T\alpha_1' + e_{+}^T\alpha_3'\right) \end{aligned} \tag{8.29}$$

Once the distances has been obtained, for a new customer $x_j \in \mathfrak{R}^n$, the attrition prediction could be obtained according to the closer hyperplane in Eq. (8.21), such as

$$f(x) = argmin\left|f_{\pm}\left(x_j\right)\right| = argmin\left|\omega_{\pm}^T\cdot x_j + b_{\pm}\right|, \tag{8.30}$$

where $|\cdot|$ means the perpendicular distance from point x_j to hyperplane $\omega_{\pm}^T\cdot x + b_{\pm} = 0$.

8.3.2 Customer Churn Prediction with NPSVM

Missing items in data would produce big problems for calculation. Usually, the missing elements are filled by some fixed real number which is easily distinguishable. Another way to process missing values is to remove the features that the ratio of missing items is higher than certain threshold (like 30%). However, this kind of operations may be subjective and not suitable.

Instead of directly deleting features, feature selection strategy is applied. Furthermore, to eliminate useless descriptors, it focuses on the relationship and missing ratio among features. Pairwise relationship between features are applied to reduce the calculation problem from the missing value. The MIC relationship measure are calculated through the available values at the same instance for each pair of features, e.g., there are five customers with two features, "-" represents missing items.

The values for the first feature are {1, 3, 7, 9, -}, and for the other are {-, 2, 6, 7, -}. So items {3, 7, 9} and {2, 6, 7} are the available values that could be used for the MIC calculation. Thus, even there are numerous missing items for some feature, the relationship between two features could also been obtained.

In order to combine feature relationship and the missing ratio together for feature filter, a new measure is defined as:

$$Preference(i) = \mathrm{Max}(MICValue) + \frac{\lambda}{\log\left(MissingRatio(i)\right) + \epsilon}, \tag{8.31}$$

where ϵ is a very small real number and λ is the parameter which is also a real number. Equation (8.31) provides a new preference measure which balances the consideration between missing ratio and relationship of features. On the left part of Eq. (8.31), *Max* (*MICValue*) is the maximum of all the MIC values among each pair of features. The right part represents the missing ratio of feature i. When affinity propagation algorithm takes this measure as parameter, as it is showed in Ref. 1, the larger the preference parameters are, the more probability an instance tended to become the center of clusters. That means when the missing ratio for feature i is too large, the corresponding value would be much smaller than the other. As a result, feature i would has less probability to be chosen as centroid. Finally, only the centroid features are preserved as the selected features.

Based on these chosen data, two hyperplane for churn and not churn customer could be constructed according to NPSVM model (8.24) and (8.26). In intuition, each hyperplane represents one category of customers. Once the two hyperplanes have been achieved, different decision functions could be obtained by providing different weights for the two distances $f_{-}(x)$ and $f_{+}(x)$. The parameter μ balances the two distances after the model construction. This characteristic provides the capability to make further adjustments when the preference has changed. The advantage is that the reconstruction and calculation for the model does not need any more. Once the two hyperplanes have been received, the further adjustments could be achieved for giving different μ values and recalculating only Eq. (8.30). The detail of the procedure could be found in Algorithm 8.3.

Algorithm 8.3 MICAP-NPSVM Customer Churn Prediction Framework

```
Input:
    Customer churn training dataset $\mathcal{D} = \{\Omega, C\}$, $\Omega = (f_1, f_2, \ldots, f_n)$. The missing ratio vector
    $MissingRatio$ for all the features. Parameters $c_1, c_2, c_3, c_4, \lambda, \mu, \epsilon$.
Output:
    Customer churn prediction function $F(x)$.
Begin
$\Omega' = \varnothing$;
for all $f_i, f_j \in \Omega, i \neq j$ do
    $\left(f_i', f_j'\right) = FilterMissingItems\left(f_i, f_j\right)$;
    $M(i, j) = CalculateMIC\left(f_i', f_j'\right)$;
end for
$M(i, i) = \mathrm{Max}(M) + \frac{\lambda}{\log(MissingRatio(i))+\epsilon}$;
$\Upsilon = \{Cluster_1, Cluster_2, \ldots, Cluster_\ell, \} = \mathrm{APClustering(M)}$;
for all $Cluster_k \in \Upsilon$ do
$I = ClusterCentroid(Cluster_k), I \in Cluster_k$;
    $\Omega' = \Omega' \cup I$
end for
Constructing new dataset $\mathcal{D}'$ according to $\Omega'$;
Generating churn customer matrix $A$ and not churn customer matrix $B$ from the new dataset $\mathcal{D}'$;
Construct and solve the optimization problem (8.24) and (8.26);
```

```
16: Calculating the distance from instance $x$ to the two hyperplanes $f_-(x)$ and $f_-(x)$ by Eqs. (8.28) and (8.29);
17: Obtain decision function by $F(x) = abs(f_-(x)) - \mu \cdot abs(f_+(x))$.
18: End
```

The input of the method is the original dataset of customer churn and the model parameters. In practice, the parameter of c_1 to c_4 could be arranged as $c_1 = c_3$, $\lambda = 0.5$, $\mu = 1$, $\epsilon = 0.001$. Line 4 extracts two different features f_i, f_j from the original dataset $\mathcal{D}$, and filters the rows with missing values in either of the two features. Line 5 calculates the MIC based on the result of line 4 and stores the MIC values in the similarity matrix M. According to Eq. (8.31), line 7 combines the maximum of all the MIC values and the missing ratio of each feature as the preference measure. Line 8 applies affinity clustering on the similarity matrix M and obtains feature cluster set Y. Line 9 to 12 selects the centroid of each cluster from Y and generates a new dataset according to the chosen features $\mathcal{D}'$ in line 13. Line 14 extracted the churn customer data as matrix A, not churn customer data as matrix B. Line 15 constructs the optimization problem (8.24) and (8.26) through matrix A and B, and then solves it. Line 16 constructs the two hyperplanes by Eqs. (8.28) and (8.29). Line 17 achieve the customer churn prediction decision function through $F(x) = abs(f_-(x)) - \mu \cdot abs(f_+(x))$. The detailed data analysis based on a well-known commercial bank of China can be found in [3].

8.4 Node-Coupling Clustering Approaches for Link Prediction

This section provides two novel node-coupling clustering approaches and their extensions for the link prediction problem. They consider the different roles of nodes, and combine the coupling degrees of the common neighbor nodes of a predicted node-pair with cluster geometries of nodes. Our approaches remarkably outperform the existing methods in terms of efficiency accuracy and effectiveness.

8.4.1 Preliminaries

8.4.1.1 Clustering Coefficient

In graph theory, clustering coefficient is a metric that can evaluate the extent to which nodes tend to cluster together in a graph [32]. It can capture the clustering information of nodes in a graph [33]. An undirected network can be described as a graph $G = (V, E)$, where V denotes the set of nodes and E indicates the set of edges. vi 2 V is a node in Graph G. The clustering coefficient of node vi in Graph G can be defined as

$$C(i) = \frac{E_i}{\frac{k_i \cdot (k_i - 1)}{2}} = \frac{2 \cdot E_i}{(k_i \cdot (k_i - 1))} \tag{8.32}$$

where $C(i)$ denotes the clustering coefficient value of node v_i. k_i represents the degree value of node v_i. E_i is the number of the con-nected links among k_i neighbors of node v_i. For example, there is a node v_1 in Graph G. The degree value of node v_1 is 5 (i.e. $k_1 = 5$). The number of connected links among the neighbors of node v_1 is 6 (i.e. $E_1 = 6$). Thus, the clustering coefficient value of node v_1 is: $C(1) = \frac{2 \cdot E_1}{(k_1 \cdot (k_1 - 1))} = \frac{2 \cdot 6}{5 \cdot (5-1)} = 0.6$.

8.4.1.2 Evaluation Metrics

In this section, we present two popular metrics for link prediction accuracy: *Area under curve (AUC)* and *Precision*. In general, a link prediction method can compute a score S_{xy} for each unknown link to evaluate its existence probability and give an ordered list of all unknown links based on these S_{xy} values [34].

It can evaluate the overall performance of a link prediction method. As described in [35, 36], the AUC value can be considered as the probability that the S_{xy} value of an existing yet unknown link is more than that of a non-existing link at random. That is, we randomly select an existing yet unknown link in the test set and compare its score with that of a non-existing link at a time. There are N independent comparisons, where the times that the existing yet unknown links have higher S_{xy} value are H, and the times that they have the same S_{xy} value are E. The *AUC* value is defined as:

$$AUC = \frac{H + 0.5 \cdot E}{N} \tag{8.33}$$

This metric considers N links with the highest S_{xy} values in all unknown links. If there are T existing yet unknown links in the top N unknown links [35, 36], the Precision is defined as:

$$Precision = \frac{T}{N} \tag{8.34}$$

8.4.2 Node-Coupling Clustering Approaches

In this section presents the proposed approaches for link prediction. Firstly, it presents a new node-coupling degree metric—node-coupling clustering coefficient. Then, it discusses the process of our approaches. Finally, the complexity analysis is given.

8.4.2.1 Node-Coupling Clustering Coefficient

Many similarity-based methods only consider the number or degrees of common neighbor nodes of a predicted node-pair in link prediction, and few exploit further the coupling degrees among the common neighbor nodes and the clustering information to improve the prediction accuracy. Based on the above reason, we propose a new node-coupling degree metric—node-coupling clustering coefficient (NCCC), which can capture the clustering information of a network and evaluate the coupling degrees between the common neighbor nodes of a predicted node-pair. It also considers different roles of the common neighbor nodes of a predicted node-pair in a network. Now, we introduce this metric through a simple example.

Figure 8.5 shows an example for predicting the link between nodes M and N in two networks. Two original networks are described in Fig. 8.5a, b. Figure 8.5c, d are two subgraphs that consist of nodes M; N and their common neighbors in Fig. 8.5a, b, respectively. We aim to predict which link between nodes M and N is more likely to exist in Fig. 8.5a, b. In general, we find that the coupling degrees of nodes M; N and their common neighbors are higher in Fig. 8.5c, d. Thus, we believe the link of nodes M; N in Fig. 8.5a is more likely to exist than in Fig. 8.5b. If we apply CN; AA; RA; PA to predict the link of nodes M; N in these two original networks, we can gain the same prediction result for each method. The reasons are as follows: from Fig. 8.5a, b, we can see that the common neighbor set {*bdf*} of nodes *M; N* are the same, and every corresponding common neighbor node has the same degree value in these two original networks. The similarity metric is the number of the common neighbor nodes of a predicted node-pair in *CN*. *CN* has the same prediction results because of the same common neighbor node set {*bdf*} of nodes M; N in these two original networks. *RA; AA* are based on the degree values of the common neighbor nodes. *RA* has the same prediction probability as *AA* because of the same degree value of every corresponding common neighbor node in {*bdf*} in these two original networks. For the same reason, *PA* provides the same prediction result because that there are the corresponding same degree values for nodes *M* and *N* in these two original networks. However, the link probabilities between node *M* and node *N* in Fig. 8.5a, b are not likely to be the same.

In the above case, inspired by [37, 38], we propose a new node-coupling degree metric based on the clustering information and node degree—node-coupling

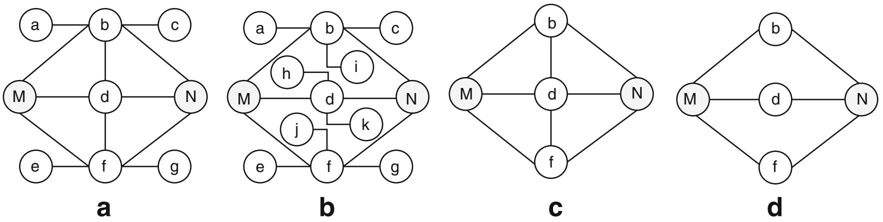


Fig. 8.5 An example for predicting the link between nodes M and N in two original networks

clustering coefficient. This metric cannot only resolve the above prediction problem in Fig. 8.5, but also capture the clustering information of a network. If node n is a com-mon neighbor node of the predicted node-pair (M, N), the node-coupling clustering coefficient of node$n$, $NCCC(n)$, can be defined as follows:

$$NCCC(n) = \frac{\sum_{i \in C_n} \left(\frac{1}{d_i} + C(i) \right)}{\sum_{j \in \Gamma_n} \left(\frac{1}{d_j} + C(j) \right)} \tag{8.35}$$

where Γ_n is the neighbor node set of nodes n. (M, N) denotes a predicted node-pair $n \in \Gamma(M) \cup \Gamma(N)$. C_n denotes the common neighbor node set of the node-pair (M, N) in $\Gamma(N)$, which includes nodes M, N. Namely $C_n = \Gamma(M) \cap \Gamma(N) \cap \Gamma(n) \cup \{M, N\}$. d_i denotes the degree value of node i. $C(i)$ denotes the clustering coefficient of node i. In this metric, $\frac{1}{d_i} + C(i)$ is considered as the contribution of node i to the coupling degree of the common neighbor nodes of the predicted node-pair (M, N). The node-coupling clustering coefficient of node n is the ratio of the contribution sum of all nodes in C_n to that in $\Gamma(n)$. In this way, our approaches can apply this metric that incorporates the clustering information and different roles of each related node to improve the prediction accuracy for link prediction.

In Eq. (8.35), since $C_n \subseteq \Gamma(n)$, $\sum_{i \in C_n} \left(\frac{1}{d_i} + C(i) \right) \leq \sum_{j \in \Gamma_n} \left(\frac{1}{d_j} + C(j) \right)$. As a result, $NCCC(n) \in (0, 1]$. Specially, $NCCC(n) = 1$ when $C_n = \Gamma(n)$.

8.4.2.2 Node-Coupling Clustering Approach Based on Probability Theory (NCCPT)

From probability theory, we propose a new link prediction approach based on the node-coupling clustering coefficient (NCCC) in the previous section. Given a pair of predicted nodes (x, y), node n is a common neighbor node of the node-pair (x, y). $P(n)$ denotes the link existence probability that node x and node y connect because of node n. $\overline{P(n)}$ denotes the link non-existence probability that node n connects node x to node y. Therefore, $P(n) = NCCC(n)$ and $\overline{P(n)} = 1 - NCCC(n)$. $\{A_1, A_2, \ldots, A_i, \ldots, A_m\}$ is the common neighbor set of the predicted node-pair (x, y), namely $\Gamma(x) \cap \Gamma(y) = \{A_1, A_2, \ldots, A_i, \ldots, A_m\}$. We assume that these common neighbor nodes of the node-pair (x, y) are independent to each other. If there exists a link between nodes x and y, at least one common neighbor node in $\{A_1, A_2, \ldots, A_i, \ldots, A_m\}$ connects node x to node y. According to probability theory, the link existence probability of the predicted node-pair (x, y), S_{xy} can be written as follows:

$$\begin{aligned} S_{xy} &= 1 - \overline{P\left(A_1\right)}\cdot \overline{P\left(A_2\right)} \cdots \overline{P\left(A_i\right)} \cdots \overline{P\left(A_m\right)} \\ &= 1-\left(1-P\left(A_1\right)\right)\cdot\left(1-P\left(A_2\right)\right)\cdots\left(1-P\left(A_i\right)\right)\cdots\left(1-P\left(A_m\right)\right) \\ &= 1-\left(1-NCCC\left(A_1\right)\right)\cdot\left(1-NCCC\left(A_2\right)\right)\cdots\left(1-NCCC\left(A_i\right)\right)\cdots \\ &\qquad \left(1-NCCC\left(A_m\right)\right) \\ &= 1-\prod\nolimits_{n\in\Gamma(x)\cap\Gamma(y)}\left(1-\frac{\sum_{i\in C_n}\left(\frac{1}{d_i}+C(i)\right)}{\sum_{j\in\Gamma_n}\left(\frac{1}{d_j}+C(j)\right)}\right) \end{aligned} \tag{8.36}$$

Equation (8.36) is a new node similarity metric in our approach. Clearly, a larger value of $\mathrm{S_{xy}}$ means a higher probability that there exists a potential link between node x and y. The related parameters in Eq. (8.36) have been described in the last section. In Eq. (8.36), since $NCCC(n) \in (0, 1]$, we will have $\prod_{n\in\Gamma(x)\cap\Gamma(y)}\left(1-\frac{\sum_{i\in C_n}\left(\frac{1}{d_i}+C(i)\right)}{\sum_{j\in\Gamma_n}\left(\frac{1}{d_j}+C(j)\right)}\right) \in (0,\,1]$, and $S_{xy} \in (0, 1]$. Specially, $S_{xy} = 1$ when $C_n = \varGamma(n)$ for every node in $\varGamma(x) \cap \varGamma(y)$. For example, we apply our approach to predict the link probability of nodes *M, N* in Fig. 8.5.

$$Fig.5(a): S_{MN} = 1-\left(1-\tfrac{2.92}{4.92(b)}\right)\cdot\left(1-\tfrac{2.8}{2.8(d)}\right)\cdot\left(1-\tfrac{2.92}{4.92(f)}\right) = 1.$$
$$Fig.5(b): S_{MN} = 1-\left(1-\tfrac{0.67}{3.67(b)}\right)\cdot\left(1-\tfrac{0.67}{2.67(d)}\right)\cdot\left(1-\tfrac{0.67}{3.67(f)}\right) = 0.50.$$

From the above computing results, we find that the potential link between node *M* and node *N* is more likely to exist in Fig. 8.5a than in Fig. 8.5b. Algorithm 8.4 describes the process of our above approach.

8.4.3 Node-Coupling Clustering Approach Based on Common Neighbors (NCCCN)

The traditional CN method is based on the number of the com-mon neighbor nodes of a predicted node-pair [39]. Its similarity metric is defined as follows:

$$S_{xy}^{CN} = \left|\Gamma(x)\cap\Gamma(y)\right| \tag{8.37}$$

where $\varGamma(i)$ denotes the common neighbor node set of node *i*. $|\varGamma(i)|$ represents the number of the common neighbor nodes of node *i*.

Although *CN* has low complexity in the link prediction problem, it does not consider the different roles of the common neighbor nodes of a predicted node-pair. This results in low prediction accuracy. Here, we propose a new link prediction approach based on *CN*, which combines the different contributions of different nodes to the connecting probability with the clustering information of a network.

In our approach, (x, y) is a predicted node-pair. Node n is a common neighbor node of the node-pair (x, y). $NCCC(n)$ can be considered as the contribution of node n that connects node x to node y. $Score(n)$ denotes the contribution score value that node n connects node x to node y. Therefore, $Score(n) = NCCC(n)$. $\{A_1, A_2, \ldots, A_i, \ldots, A_m\}$ is the common neighbor set of the predicted node-pair (x, y), namely $\varGamma(x) \cap \varGamma(y) = \{A_1, A_2, \ldots, A_i, \ldots, A_m\}$. Here, these common neighbor nodes of the node-pair (x, y) are assumed to be independent to each other. We use the contribution sum of all common neighbor nodes of the predicted node-pair (x, y), S_{xy}, to evaluate the link existence likelihood between node x and y. Therefore, the new similarity metric in our approach is defined as follows:

$$\begin{aligned} S_{xy} &= Score\,(A_1) + Score\,(A_2) + \cdots + Score\,(A_i) + \cdots + Score\,(A_m) \\ &= NCCC\,(A_1) + NCCC\,(A_2) + \cdots + NCCC\,(A_i) + \cdots + NCCC\,(A_m) \\ &= \textstyle\sum_{1 \le i \le m} NCCC\,(A_i) \\ &= \textstyle\sum_{n \in \Gamma(x) \cap \Gamma(y)} \frac{\sum_{i \in C_n}\left(\frac{1}{d_i} + C(i)\right)}{\sum_{j \in \Gamma_n}\left(\frac{1}{d_j} + C(j)\right)} \end{aligned} \tag{8.38}$$

In our approach, the related parameters in Eq. (8.38) have been described in Sect. 8.4.2. Clearly, a larger value of S_{xy} means a higher likelihood that there exists a potential link between node x and y. $\varGamma(x) \cap \varGamma(y)$ is the number of common neighbor nodes of the predicted node-pair (x, y). In Eq. (8.38), since $0 < NCCC(n) \le 1$, we have $0 < \sum_{n \in \Gamma(x) \cap \Gamma(y)} \frac{\sum_{i \in C_n}\left(\frac{1}{d_i} + C(i)\right)}{\sum_{j \in \Gamma_n}\left(\frac{1}{d_j} + C(j)\right)} \le \mid \Gamma(x) \cap \Gamma(y) \mid$, and $S_{xy} \in (0, \mid \varGamma(x) \cap \varGamma(y) \mid)$. Specially, $S_{xy} = \mid \varGamma(x) \cap \varGamma(y) \mid$ when $C_n = \varGamma(n)$ or every node in $\varGamma(x) \cap \varGamma(y)$. For example, we use this approach to compute the similarity score of the predicted node-pair (M, N) in Fig. 8.5a, b, respectively.

$$\begin{aligned} Fig.5(a) : S_{MN} &= \tfrac{2.92}{4.92(b)} + \tfrac{2.8}{2.8(d)} + \tfrac{2.92}{4.92(f)} = 2.19. \\ Fig.5(b) : S_{MN} &= \tfrac{0.67}{3.67(b)} + \tfrac{0.67}{2.67(d)} + \tfrac{0.67}{3.67(f)} = 0.61. \end{aligned}$$

We obtain the same prediction result as NCCPT. From this example, we find that our node-coupling clustering approaches can provide better prediction results than the traditional methods. Algorithm 8.4 illustrates the process of our above approach.

8.4.4 The Extensions of NCCPT and NCCCN

To further improve the performance of link prediction, we extend NCCPT and NCCCN by adding its clustering coefficient information of every selected common neighbor node, C_n, to the above two approaches respectively.

For the same reason, (x, y) is a predicted node-pair; node n is a common neighbor node of the node-pair (x, y). When we add the node clustering coefficient information, C_n, in the contribution of node n to the connecting probability based on NCCPT, we can obtain a new contribution of node n:

$NCCC(n) + C_n$. However, $0 \leq NCCC(n) + C_n \leq 2$. This is outside the scope of the probability value. In order to extend NCCPT, we use the average value of $NCCC(n)$ and C_n, $\frac{1}{2}\cdot (C_n + NCCC(n))$, as the contribution of node n. Therefore, S_{xy} in the extended NCCPT approach (ENCCPT) is defined as follows:

$$\begin{aligned} S_{xy} &= 1 - \overline{P(A_1)}\cdot\overline{P(A_2)}\cdots\overline{P(A_i)}\cdots\overline{P(A_m)} \\ &= 1-(1-P(A_1))\cdot(1-P(A_2))\cdots(1-P(A_i))\cdots(1-P(A_m)) \\ &= 1-\left(1-\tfrac{1}{2}\cdot(NCCC(A_1)+C(A_1))\right)\cdot\left(1-\tfrac{1}{2}\cdot(NCCC(A_2)+C(A_2))\right) \\ &\cdots\left(1-\tfrac{1}{2}\cdot(NCCC(A_i)+C(A_i))\right)\cdots\left(1-\tfrac{1}{2}\cdot(NCCC(A_m)+C(A_m))\right) \\ &= 1-\prod\nolimits_{n\in\Gamma(x)\cap\Gamma(y)}\left(1-\tfrac{1}{2}\cdot\left(\frac{\sum_{i\in C_n}\left(\frac{1}{d_i}+C(i)\right)}{\sum_{j\in\Gamma_n}\left(\frac{1}{d_j}+C(j)\right)}\right)+C_n\right) \end{aligned} \tag{8.39}$$

where C_n is the clustering coefficient of node n. The other parameters are the same as Eq. (8.36). In Eq. (8.39), since $NCCC(n) \in (0, 1]$ and $C_n \in [0, 1]$, we have $\prod_{n\in\Gamma(x)\cap\Gamma(y)}\left(1-\frac{1}{2}\cdot\left(\frac{\sum_{i\in C_n}\left(\frac{1}{d_i}+C(i)\right)}{\sum_{j\in\Gamma_n}\left(\frac{1}{d_j}+C(j)\right)}\right)+C_n\right) \in [0, 1)$, and $S_{xy} \in (0, 1]$. Specially, $S_{xy} = 1$ when $C_n = \Gamma(n)$ and $C_n = 1$ for every node in $\Gamma(x) \cap \Gamma(y)$. For instance, ENCCPT is used to predict the link existence probabilities of node-pair (M, N) in Fig. 8.5a, b as follows:

$$\begin{aligned} Fig.5(a): S_{MN} &= 1-\left(1-0.5\cdot\left(\tfrac{2.92}{4.92}+0.2\right)(b)\right)\cdot\left(1-0.5\cdot\left(\tfrac{2.8}{2.8}+0.67\right)(d)\right) \\ &\cdot\left(1-0.5\cdot\left(\tfrac{2.92}{4.92}+0.2\right)(f)\right)=0.94. \\ Fig.5(b): S_{MN} &= 1-\left(1-0.5\cdot\left(\tfrac{0.67}{3.67}+0\right)(b)\right)\cdot\left(1-0.5\cdot\left(\tfrac{0.67}{2.67}+0\right)(d)\right) \\ &\cdot\left(1-0.5\cdot\left(\tfrac{0.67}{3.67}+0\right)(f)\right)=0.28. \end{aligned}$$

Similarly, (x, y) represents a pair of predicted nodes, and node n is a common neighbor node of the node-pair (x, y). When we add the node clustering coefficient information, C_n, in the contribution of node n that connects node x to node y based on NCCCN, we can obtain a new contribution of node n: $NCCC(n) + C_n$. $Score(n)$ denotes the contribution score value that node n connects node x to node y. Therefore, $Score(n) = NCCC(n) + C_n$. Hence, the extended NCCCN approach (ENCCCN) is shown in the following Eq. (8.40).

$$\begin{aligned} S_{xy} &= Score\,(A_1)+Score\,(A_2)+\cdots+Score\,(A_i)+\cdots+Score\,(A_m)\\ &= (NCCC\,(A_1)+C\,(A_1))+(NCCC\,(A_2)+C\,(A_2))+\cdots\\ &\quad+(NCCC\,(A_i)+C\,(A_i))+\cdots+(NCCC\,(A_m)+C\,(A_m))\\ &=\textstyle\sum_{1\le i\le m}(NCCC\,(A_i)+C\,(A_i))\\ &=\textstyle\sum_{n\in\Gamma(x)\cap\Gamma(y)}\left(\frac{\sum_{i\in C_n}\left(\frac{1}{d_i}+C(i)\right)}{\sum_{j\in\Gamma_n}\left(\frac{1}{d_j}+C(j)\right)}+C(n)\right)\end{aligned} \tag{8.40}$$

where $C(n)$ denotes the clustering coefficient of node n. Other parameters are the same as Eq. (8.38). In Eq. (8.40), since $0 < NCCC(n) \le 1$ and $0 \le C(n) \le 1$, we have $0 < NCCC(n) + C(n) \le 2$, and $0 < \sum_{n\in\Gamma(x)\cap\Gamma(y)}\left(\frac{\sum_{i\in C_n}\left(\frac{1}{d_i}+C(i)\right)}{\sum_{j\in\Gamma_n}\left(\frac{1}{d_j}+C(j)\right)}+C(n)\right) \le$ $2\cdot|\ \Gamma(x)\cap\Gamma(y)\ |$, namely $S_{xy} \in (0, 2\cdot\ |\Gamma(x)\cap\Gamma(y)|\ ]$. Specially, $S_{xy} = 2\cdot\ |\Gamma(x)\cap\Gamma(y)|$ when $C(n) = \Gamma(n)$ and $C(n) = 1$ for every mode in $\Gamma(x)\cap\Gamma(y)$. For instance, we use ENCCCN to predict the existence possibility of the link between nodes M and N in Fig. 8.5a, b, respectively. The results are as follows:

$$\begin{aligned} Fig.5(a): S_{MN} &= 1-\left(1-0.5\cdot\left(\tfrac{2.92}{4.92}+0.2\right)(b)\right)\cdot\left(1-0.5\cdot\left(\tfrac{2.8}{2.8}+0.67\right)(d)\right)\\ &\quad\cdot\left(1-0.5\cdot\left(\tfrac{2.92}{4.92}+0.2\right)(f)\right)=0.94.\\ Fig.5(b): S_{MN} &= 1-\left(1-0.5\cdot\left(\tfrac{0.67}{3.67}+0\right)(b)\right)\cdot\left(1-0.5\cdot\left(\tfrac{0.67}{2.67}+0\right)(d)\right)\\ &\quad\cdot\left(1-0.5\cdot\left(\tfrac{0.67}{3.67}+0\right)(f)\right)=0.28.\end{aligned}$$

From the above prediction results, it can be found that ENCCPT and ENCCCN have the same prediction results as NCCPT and NCCCN. Moreover, we notice that the prediction results of ENCCPT and ENCCCN have more obvious differences than NCCPT and NCCCN in the same example, respectively. This results in better prediction results compared with our baseline approaches (i.e. NCCPT, NCCCN), and it shows the importance of the clustering information in the link prediction. Algorithm 8.4 describes the process of our above extended approaches.

8.4.5 Complexity Analysis of Our Approaches

In real applications, most link prediction methods are based on local analysis and global analysis. *CN* is the simplest link prediction method in these methods. As a representative of the methods based on local analysis, *CN* has low complexity and suitable for large-scale networks. Its time complexity is $O(n^2)$, where n is the number of nodes in a network. Its space complexity is $O(n^2)$. In contrast, *Katz* is a representative of the methods based on global analysis. Its time complexity is $O(n^3)$. Its space complexity is $O(n^2)$. The methods based on global analysis are impractical for large-scale networks because of their high complexity.

Algorithm 8.4 Node-Coupling Clustering Approaches

```
 1: Set $d[\,] = 0$; $C[\,] = 0$;
 2: Divide the original network $G$ into the training set $TS$ and test set $PS$;
 3: for each node $i$ in $G$ do
 4:     Compute the degree value of this node: $d[i]$;
 5:     Compute the clustering coefficient of this node: $C[i]$;
 6: end for
 7: for each nonexistent edge $(x, y)$ in $G$ do
 8:     Compute the similarity score $S_{xy}$ by Eqs. (8.36), (8.38), (8.39) or Eq. (8.40);
 9: end for
10: Arrange the list of all $\mathrm{S_{xy}}$ in descending order;
11: Compute AUC by Eq. (8.33);
12: Return $AUC$;
```

As illustrated in Algorithm 8.4, the main operations of our algorithms consist of lines 3–6 and lines 7–9. The time complexity of lines 3–6 is Oðn^{2}Þ in the worst case. The time complexity of lines 7–9 is $O(n^2)$. Therefore, the overall time complexity of our algorithms is $O(n^2)$. The space complexity of our algorithms is $O(n^2)$. Because our approaches have the same complexity as *CN*, they are suitable for large-scale networks. The experimental analysis to evaluate the performance of proposed approaches on two synthetic datasets and six real datasets can be found in [4].

8.5 Pyramid Scheme Model for Consumption Rebate Frauds

This section provides a pyramid scheme model which has the principal characters of many pyramid schemes that have appeared in recent years: promising high returns, rewarding the participants for recruiting the next generation of participants, and the organizer takes all of the money away when they find that the money from the new participants is not enough to pay the previous participants interest and rewards. It assumes that the pyramid scheme is carried out in the tree network, **Erdős-Réney** (ER) random network, Strogatz–Watts (SW) small-world network, or Barabasi–Albert (BA) scale-free network. The section then gives the analytical results of the generations that the pyramid scheme can last in these cases.

8.5.1 Networks

8.5.1.1 Tree Network

Tree networks are connected acyclic graphs. The word tree suggests branching out from a root and never completing a cycle. Tree networks are hierarchical, and each node can have an arbitrary number of child nodes. Trees as graphs have many applications, especially in data storage, searching, and communication [40].

8.5.1.2 Random Network

Random network, also known as stochastic network or stochastic graph, refers to a complex network created by stochastic process. The most typical random network is the ER model proposed by Paul **Erdős** and Alfred **Réney** [41]. The ER model is based on a natural construction method: suppose there are n nodes, and assume that the possibility of connection between each pair of nodes is constant 0 < p < 1. The network constructed in this way is an ER model network. Scientists first used this model to explain real-life networks.

8.5.1.3 Small-World Network

The original model of small-world was first proposed by Watts and Strogatz, and it is the most classical model of small-world network, which is called SW small-world network [32]. The SW small-world network model can be constructed as follows: take a one-dimensional lattice of L vertices with connections or bonds between nearest neighbors and periodic boundary conditions (the lattice is a ring), then go through each of the bonds in turn and independently with some probability ϕ "rewiring" it. Rewiring in this context means shifting one end of the bond to a new vertex chosen uniformly at random from the whole lattice, with the exception that no two vertices can have more than one bond running between them and no vertex can be connected by a bond to itself. The most striking feature of small-world networks is that most nodes are not neighbors of one another, but the neighbors of any given node are likely to be neighbors of each other and most nodes can be reached from every other node by a small number of hops or steps. It has been found that many networks in real life have the small-world property, such as social networks [42], the connections of neural networks [32], and the bond structure of long macromolecules in the chemical [43].

8.5.1.4 Scale-Free Network

A scale-free network is a network whose degree distribution follows a power law, at least asymptotically. The first model of scale-free network was proposed by

Barabasi and Albert, which is called BA scale-free network [44]. The BA model describes a growing open system starting from a group of core nodes, new nodes are constantly added to the system. The two basic assumptions of the BA scale-free network model are as follows: (1) from m_0 nodes, a new node is added at each time step, and m nodes from the m_0 nodes are selected to be connected to the new node ($m \leq m_0$); (2) the probability $\mathit{\Pi}_i$ that the new node is connected to an existing node i satisfies $\Pi_i = k_i / \sum_{j=1}^{N-1} k_j$, where k_i denotes the degree of the node i and N denotes the number of nodes. In this way, when added enough new nodes, the network generated by the model will reach a stable evolution state, and then the degree distribution follows the power law distribution. In [45], it was shown that the degree distribution of many networks in real world is approximate or exact obedience to the power law distribution.

8.5.2 *The Model*

8.5.2.1 Assumptions

We consider a simple pyramid scheme that meets the basic features of many pyramid schemes in the real world, especially the consumption rebate platforms. First, it has an organizer that attracts participants through promising a high rate of return compared to the normal interest rate. Besides the promising return, any participant will be rewarded by the organizer with a proportion of the total investment of the participants he or she directly attracted, thus the early participants will be motivated enough to recruit the next-generation participants and the next-generation participants will do the same thing in order to get more returns. Secondly, we assume all the participants at current generation are recruited by the participants at the upper generation, and the organizer pays the participants at the previous generations the interests and rewards when all possible participants at current generation have joined in the scheme. The third assumption is that the organizer will take all the money away when he finds the money from the new participants is not enough to pay the previous participants interest and rewards. To simplify the model, we also assume all the participants invest the same amount of money and invest only once. Figure 8.6 is a schematic diagram of the pyramid scheme, which has one organizer and two generations of participants.

Based on these assumptions, we discuss the pyramid scheme spreads in the tree network, random network, small world network, and scale-free network below.

8.5.2.2 Tree Network Case

If the pyramid scheme expands in the form of tree network that has a constant branching coefficient a and the root node of the tree network represents the organiser, we can simply write the number of participants at the g-th generation

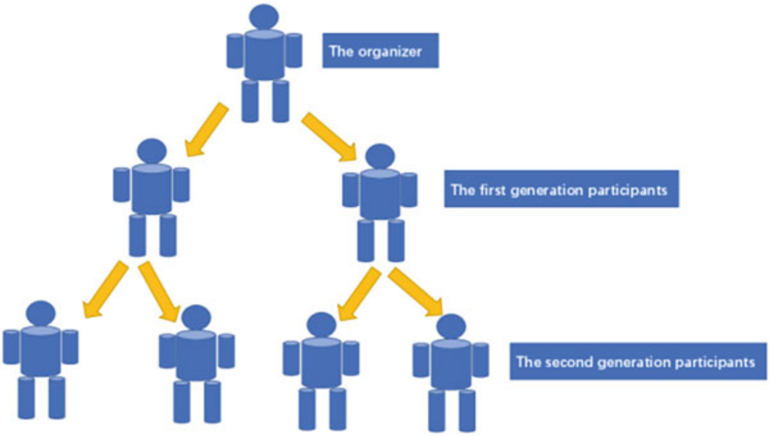

The organizer
The first generation participants
The second generation participants

Fig. 8.6 A schematic diagram of pyramid scheme. From top to bottom are the organizer, the first-generation participants, and the second-generation participants

as $n_1\ \alpha^{g\,-\,1}$ and the total amount of money entering the pyramid scheme at the g-th generation as $mn_1\ \alpha^{g\,-\,1}$, where n_1 is the number of participants at the first generation and m is the amount of money that every participant invests. For simplification, we assume $n_1 = a$ and $m = 1$. We suppose the number of all potential participants is N in this case. Removing the interest and rewards, the relationship between the net inflow of money M of the pyramid scheme and the generation g when all possible participants at the g-th generation have joined in the scheme can be given by:

$$M(g) = \alpha^g - r_0 \sum_{i=1}^{g-1} \alpha^i - r_1 \alpha^g \tag{8.41}$$

where r_0 is the promised rate of return of the organizer, and r_1 is the ratio of the money rewarded to a participant to the total investment of the participants he or she directly recruited. Normally in real pyramid scheme cases, r_0 and r_1 are between 0% and 50%. The first term of Eq. (8.41) represents the investment of all the participants, the second term represents the interest paid to the participants before the generation g, and the third term represents the rewards paid to the recruiters of participants at the g-th generation. Notice that in our pyramid scheme, the participants at the g-th generation are all recruited by the participants at the *(g − 1)*-th generation.

The second term of Eq. (8.41) is the sum of geometric sequences, after summing them up, Eq. (8.41) can be rewritten as:

$$M(g) = \frac{\alpha}{\alpha - 1}\left[(1 - r_1)\,\alpha^g - (1 + r_0 - r_1)\,\alpha^{g-1} + r_0\right]. \tag{8.42}$$

Through Eq. (8.42) we can find that if the branching coefficient a satisfies the condition:

$$\alpha \geq \frac{1 - r_1 + r_0}{1 - r_1} \tag{8.43}$$

the inflow of money M(g) of the pyramid scheme is always positive, so the pyramid scheme will continue forever under the circumstances.

However, the potential participants are limited to N and the pyramid scheme will stop eventually. The maximum generation G of the pyramid scheme is given by:

$$G_{TR} = \left[\log_{\alpha} \frac{N\alpha - N + \alpha}{\alpha} \right] + 1 \tag{8.44}$$

where $[x]$ is the integer part of x. At the G-th generation, all the potential participants have joined the pyramid scheme, and the organizer will take away all the money and not pay the interest and rewards any more. We can write the final income of the pyramid as:

$$R_p = N - r_0 \sum_{i=1}^{G-2} (G - i - 1)\, \alpha^i - r_1 \sum_{i=2}^{G-1} \alpha^i \tag{8.45}$$

and the income of the participants at the i-th generation is:

$$R_i = \begin{cases} r_0 (G - i - 1) \cdot \alpha^i + r_1 \alpha^{i+1} - \alpha^i, \quad for\ 1 \leq i \leq G - 2, \\ \qquad - \alpha^i, \quad for\ G - 2 \leq i \leq G. \end{cases} \tag{8.46}$$

Figure 8.7a shows the analytical result and the simulative result of maximum generation G_{ER} when the branching coefficient α changes, and we take the parameters $N = 10{,}000$, $r_0 = 0.1$, $r_1 = 0.1$. Figure 8.7b shows the analytical result and the simulative result of maximum generation G_{ER} when the number of possible participants N changes, and we take the parameters $\alpha = 4$, $r_0 = 0.1$, $r_1 = 0.1$. Figure 8.7 illustrates intuitively that in the tree network case, if other conditions of the pyramid scheme remain unchanged, the larger the branch coefficient—that is, the newer participants each person recruits—, the fewer generations the pyramid scheme can last. Meanwhile, when other conditions remain unchanged, the larger the number of potential participants, the more generations the pyramid scheme can sustain, but every new generation needs more participants and this growth of new participants is exponential.

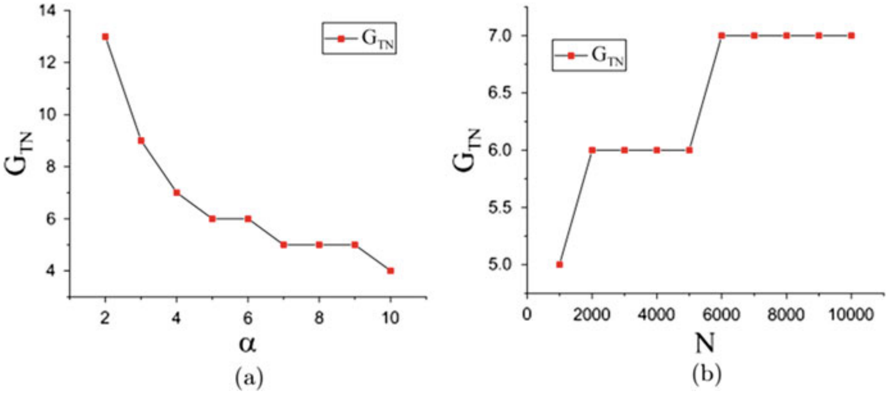


Fig. 8.7 (**a**) The analytical result and the simulative result of maximum generation G_{TN} when the branching coefficient a change. We take the parameters N = 10,000, $r_0 = 0.1$, $r_1 = 0.1$. (**b**) The analytical result and the simulative result of maximum generation G_{TN} when the number of possible participants N changes. We take the parameters $a = 4$, $r_0 = 0.1$, $r_1 = 0.1$

8.5.2.3 Random Network Case

If the pyramid scheme takes place in an ER random net-work that has an average degree k and N nodes, we assume the organizer is a random node in the network and other nodes represent the potential participants. The organizer recruits the potential participants nearest to him as the first-generation participants, and the first-generation participants recruit the potential participants nearest to them as the second-generation participants, and so on. So the generation of any participant in the pyramid scheme is given by the shortest path length to the node representing the organizer. Katzav et al. [46] have given the approximate analytical results for the distribution of shortest path lengths in ER random networks, the number of nodes at the i-th generation is about k^i if $ilog_N k < 1$ and all the nodes are included in the pyramid scheme if $ilog_N k > 1$. Therefore, the pyramid scheme in the ER random network is approximate to the case in the tree network above and the difference is the branching coefficient α should be replaced by the average degree k.

First, like the case in the tree network, r_0, r_1, and k should satisfy the following condition:

$$k \geq \frac{1 - r_1 + r_0}{1 - r_1} \tag{8.47}$$

The approximate maximum generation G of the pyramid scheme in this case is given by:

$$G_{ER} \approx \left| 1/\log_N k \right| + 1 \tag{8.48}$$

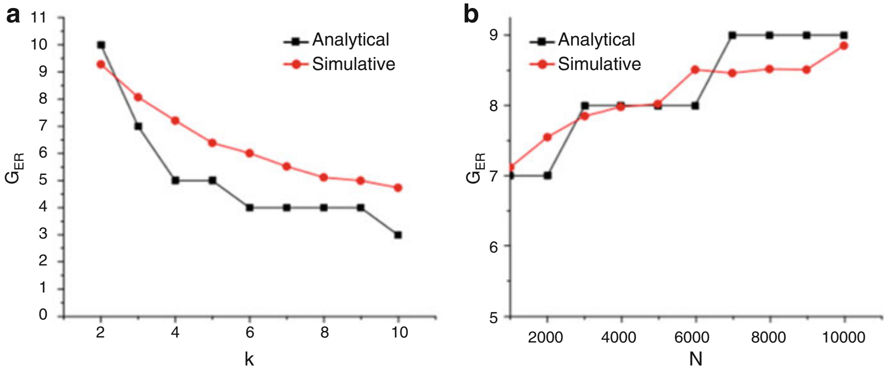


Fig. 8.8 (**a**) The analytical result and the simulative result of maximum generation G_{ER} when the average degree k changes. We take the parameters $N = 10{,}000$, $r_0 = 0.1$, $r_1 = 0.1$. (**b**) The analytical result and the simulative result of maximum generation G_{ER} when the number of possible participants N changes. We take the parameters $k = 4$, $r_0 = 0.1$, $r_1 = 0.1$. The simulative results are averaged after 100 simulations

In addition, we can also write the approximate expressions of the organiser's and participants' income which have the same form of Eq. (8.46), which we omit here. Figure 8.8a shows the analytical result and the simulative result of maximum generation G_{ER} when the average degree k changes, and we take the parameters $N = 1000$, $r_0 = 0.1$, $r_1 = 0.1$. Figure 8.8b shows the analytical result and the simulative result of maximum generation G_{ER} when the number of possible participants N changes, and we take the parameters $k = 4$, $r_0 = 0.1$, $r_1 = 0.1$. The simulative results in the figures are averaged after 100 simulations. From Fig. 8.8, we can find that in the ER random network case, the relationship between maximum generation G_{ER} and mean degree k, and the relationship between G_{ER} and N are similar to those in the tree network case, where the mean degree k represents the amount of participants that each participant can recruit averagely. We can also find that within the range of parameters we have chosen, the analytical results and simulative results are very close.

8.5.2.4 Small World Network Case

Now we consider the pyramid scheme carries out in an SW small-world network, to some extent this case is similar to the case in the ER random network. We also randomly choose a node as the organizer, other nodes represent the potential participants, and r_0, r_1 represent the interest rate and reward ratio, respectively. The generation of any participant in the pyramid scheme is the shortest path length to the node representing the organizer. Newman and Watts [47] pointed out that the number of nodes increases exponentially with the average length of the shortest path when the nodes are infinite. The approximate surface area of a sphere of radius r on the SW small-world network can be given by [47]

$$A(r) = 2e^{4r/\xi}, \tag{8.49}$$

where $\xi = 1/\phi k$, and ϕ is the rewriting probability and k is the degree of the corresponding rule graph.

Changing r to g, we can obtain the approximate number of participants at the g-th generation. Because of the exponential form of *A(g)*, we can deal with this case just like in the cases of tree network and ER random network. The branching coefficient α should be replaced by $e^{4/\xi}$, and the following condition should be satisfied:

$$e^{4/\xi} \geq \frac{1 - r_1 + r_0}{1 - r_1}. \tag{8.50}$$

If the nodes are finite, then the number of nodes reaches the maximum when the distance from the node to the organizer is near the average length of the shortest path. If the distance is greater than the average length of the shortest path, the number of nodes quickly reduces to 0, so it can be approximately considered that the maximum generation G is close to the average length of the shortest path. The average path length d of the SW small-world network is given by [47]

$$\bar{l}_{SW} \approx \frac{N}{K} f\left(\phi K N\right), \tag{8.51}$$

where

$$f(u) \approx \begin{cases} 1/4, & if\ u \to 0, \\ lnu/u, & if\ u \to \infty. \end{cases} \tag{8.52}$$

The number of nodes with the average shortest path length to the node representing the organizer is the largest. So we can infer the maximum generation G_{SW} of the pyramid scheme is given by [48]

$$G_{SW} \approx \left[\bar{l}_{SW}\right] + 1 \tag{8.53}$$

In the simulation, we find that the values of r_0 and r_1 are very important. Generally speaking, the greater the values of r_0 and r_1 satisfying Eq. (8.50) are, the closer the simulation results and numerical results are. This happens because when the values of r_0 and r_1 are larger, the pyramid scheme can easily terminate when the number of generations exceeds the average shortest path length. Figure 8.9a shows the analytical result and the simulative result of maximum generation G_{SW} when the possible participants f changes, and we take the parameters $N = 1000$, $K = 3$, $r_0 = 0.2$, $r_1 = 0.2$. Figure 8.9b shows the analytical result and the simulative result of maximum generation $\mathrm{G_{SW}}$ when the number of possible participants N changes, and we take the parameters $K = 3$, $\phi = 0.1$, $r_0 = 0.2$, $r_1 = 0.2$. The

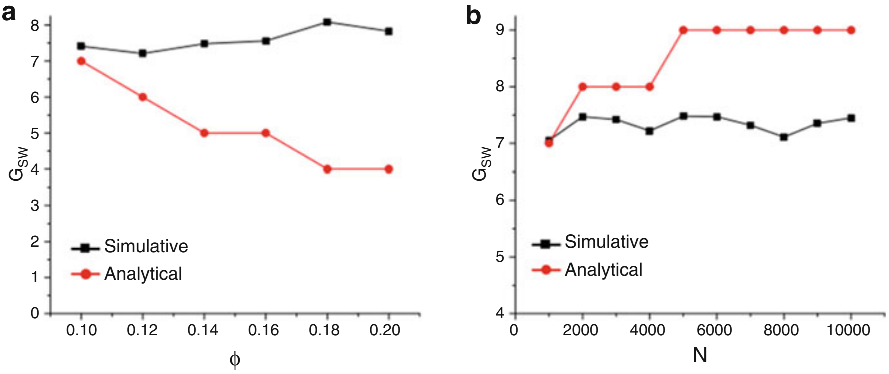


Fig. 8.9 (**a**) The analytical result and the simulative result of maximum generation G_{SW} when the possible participants f changes. We take the parameters $N = 1000$, $K = 3$, $r_0 = 0.2$, $r_1 = 0.2$. (**b**) The analytical result and the simulative result of maximum generation GSW when the number of possible participants N changes. We take the parameters $K = 3$, $\phi = 0.1$, $r_0 = 0.2$, $r_1 = 0.2$. The simulative results are averaged after 100 simulations

simulative results in the figures are averaged after 100 simulations. In Fig. 8.9, we find that within the range of parameters we selected, the maximum generation G_{SW} of the pyramid scheme is not very sensitive to the reconnection probability ϕ and the potential participants, and the analytical results are basically in accordance with the simulative results.

8.5.2.5 Scale-Free Network Case

If the pyramid scheme expands in a BA scale-free net-work, similar to the cases in ER random network and SW small-world network above, then we also randomly choose a node as the organizer and other nodes represent the potential participants. The organizer recruits participants and the participants recruit the next generation participants through the network connections. To ensure positive inflows, the following condition must be satisfied:

$$(1 - r_1)\, n\,(g + 1) \geq r_0 \sum_{i=1}^{g} n(g), \tag{8.54}$$

where *n(g)* represents the number of participants at the *g*-th generation, and *n(g + 1)* represents the number of participants at the *(g + 1)*-th generation. The distribution of shortest path length approximates the normal distribution and the position corresponding to the highest point of the normal distribution is the average shortest path length [49]. The average path length s of the BA scale-free network is given by [50]

$$\bar{l}_{BA} \approx \frac{lnN}{lnlnN}. \tag{8.55}$$

Before the peak, the number of participants per generation grows faster than the exponential growth. But after that, the number of participants per generation declines rapidly, so the condition can no longer be satisfied. So we can infer that the maximum generation G_{BA} is close to the average shortest path length, and is given by

$$G_{BA} \approx \left[\bar{l}_{BA}\right] + 1. \tag{8.56}$$

Figure 8.10 shows the analytical result and the simulative result of maximum generation $\mathrm{G_{BA}}$ when the number of possible participants N changes. We taken the parameters $r_0 = 0.2$, $r_1 = 0.2$, and the simulative result is averaged after 100 simulations. We can find that in scale-free networks, the maximum generation G_{BA} is not very sensitive to the potential participants in Fig. 8.10. The analytical results can basically reflect this characteristic.

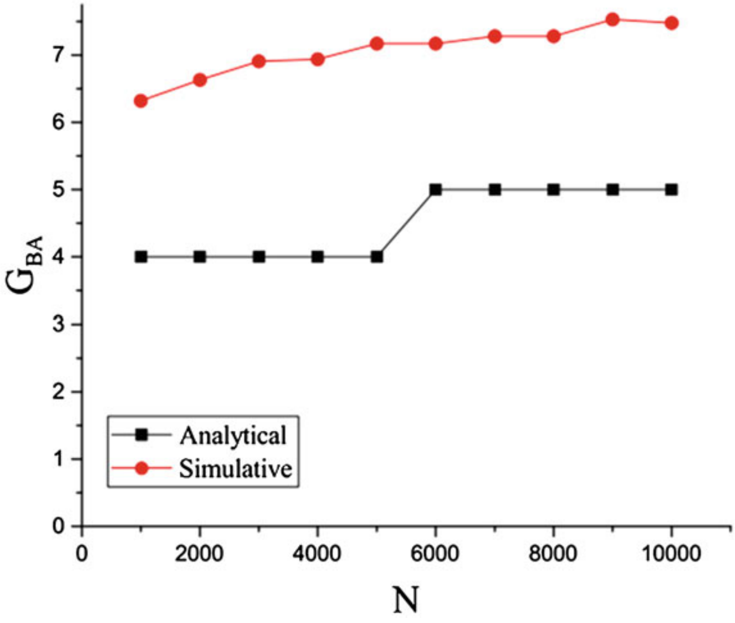


Fig. 8.10 The analytical result and the simulative result of maximum generation G_{BA} when the number of possible participants N changes. We take the parameters $r_0 = 0.2$, $r_1 = 0.2$. The simulative result is averaged after 100 simulations

8.5.3 A Pyramid Scheme in Real World

Although real cases of pyramid scheme are easy to find in news reports, there are few cases that give details of the number of people involved and the pyramid generations. Usually, when the organizer of the pyramid scheme disappears, the participants with loss will report the case to the police, who will then investigate the case. On July 23, 2018, China news network Guangzhou Station reported a pyramid scheme that had 75,663 account numbers and 46 generations, and the pyramid scheme had amassed 76 million yuan in 3 months [51]. This is the same type of pyramid scheme as described in the introduction. Using the analysis in Sect. 8.3, we assume that the pyramid scheme carries on the tree network, **ER** random network, SW small-world network, and BA scale-free net-work. We then verify which network can describe the pyramid scheme in the real world well. We assume that one account number represents a participant.

If this real pyramid scheme expands in a tree network, we can calculate the tree network' branching coefficient $\alpha \approx 1.28$. This means on average less than two participants are recruited by each participant. However, we cannot know more about the connections between the participants, except the branching coefficient.

If this real pyramid scheme spreads in an **ER** Random network, we can calculate the average degree $k \approx 1.28$ through Eq. (8.48). So each node is connected to 1.28 nodes on average, and the connection probability in the ER random network is less than $1.28/75663 \approx 1.7 \times 10^{-5}$, which is very small, then isolated nodes and nodes with degree 1 easily to appear in the network. Although this case is similar to that of tree network, the branching coefficient in the random network is not stable and it is easy to end the pyramid scheme if Eq. (8.43) is not satisfied (the minimum of the formula $(1 - r_1 + r_0)/(1 - r_1)$ is greater than 1). So we think the pyramid scheme can hardly happen in the ER random network.

If this real pyramid scheme carries out in a BA scale-free network, through the analysis and simulation, we find that developing to 46 generations needs far more than 75,663 participants. Therefore, the connections between participants are impossible to form a BA scale-free network.

If this real pyramid scheme takes place in a SW small-world network and accords to all our assumptions, then we could find a simulative result to fit the result of the real pyramid scheme. The parameters we select are $N = 100{,}000$, $\phi = 0.02$, $K = 4$, $r_0 = 0.1$, $r_1 = 0.1$, and each participant invests 23,500 yuan. The simulative pyramid scheme has 74,652 participants, and develops to 46 generations, and the fund pool of the pyramid is about 76 million yuan. The simulation results are in good agreement with the real pyramid scheme. Figure 8.11a, b show the cumulative number of participants N_{cum}, the number of participants N_{g} in each generation, and the cumulative money M_{cum} changing over generation g in the simulative pyramid scheme.

Figure 8.11 shows that, the number of participants and the amount of accumulated money of the pyramid scheme grow slowly in the initial stage and explosively in the later stage. Once the growth rate slows down, the amount of the

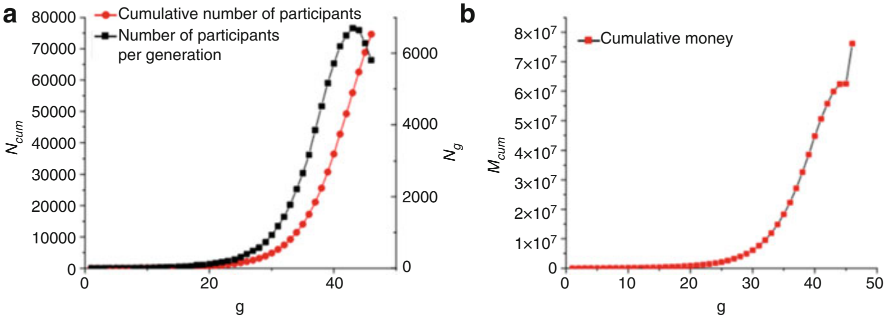


Fig. 8.11 The cumulative number of participants N_{cum}, the number of participants N_{g} in each generation, and the cumulative money M_{cum} changing over generation g in the simulative pyramid scheme. This is one simulative result in the SW small-world case, the parameters we select are $N = 100{,}000$, $\phi = 0.02$, $r_0 = 0.1$, $r_1 = 0.1$, and each participant invests 23,500 yuan

pyramid scheme's accumulated money will soon reach a peak and the organizer will escape. The probability of reconnection in simulation is 0.02, which can be understood according to the actual situation and means that participants tend to recruit new participants from familiar people. In fact, according to our investigation and many news reports, such pyramid frauds always arise in small cities and most of the participants recruit new participants from their familiar people. As the generations go on, the network constituted by all participants has the properties of small world: agglomeration and having some flocks, which are similar to the interpersonal network. Although our model has been simplified and approximated, it is enlightening to explain the real case.

Through this simulation analysis, we can speculate that the connections between participants in the real case may constitute a SW small-world network.

This work is helpful to understand the operation mechanism and characteristics of the pyramid schemes of consumption rebate type. The model may be able to apply to some current illegal high-interest loans, if these illegal projects promise a high interest rate and reward the investors who encourage others to invest in such projects but the money accumulated is not actually invested in any real projects. It shows that the pyramid schemes of consumption rebate type are not easy to be detected by the supervision because of the small amount of funds and the small number of participants accumulated in the initial stage. After the rapid growth of funds and participants, it often comes to the end of this kind of pyramid frauds, and the organizers have often already fled. Therefore for regulators, it is better to nip such platforms in the bud to avoid any more people suffering a loss. In addition, to some extent, this research finding provides some basis for further study of such frauds. For example, we will further consider how the participants' beliefs about always having enough new participants affect the operation of these frauds.

References

1. Deng, W., Shi, Y., Chen, Z., Kwak, W., Tang, H.: Recommender system for marketing optimization. World Wide Web. **23**(4), 1–21 (2020)
2. Lee, J., Shi, Y., Wang, F., Lee, H., Kim, H.K.: Advertisement clicking prediction by using multiple criteria mathematical programming. World Wide Web. **19**(4), 707–724 (2016)
3. Zhao, X., Shi, Y., Lee, J., Kim, H.K., Lee, H.: Customer churn prediction based on feature clustering and nonparallel support vector machine. Int. J. Inf. Technol. Decis. Making. **13** (2014)
4. Li, F., He, J., Huang, G., Zhang, Y., Shi, Y., Zhou, R.: Node-coupling clustering approaches for link prediction. Knowl. Based Syst. **2015**, S0950705115003536 (2015)
5. Shi, Y., Li, B., Long, W.: Pyramid scheme model for consumption rebate frauds. Chin. Phys. B. **28**(7), 078901 (2019). https://doi.org/10.1088/1674-1056/28/7/078901
6. Pinder, J.E., Wiener, J.G., Smith, M.H.: The Weibull distribution: a new method of summarizing survivorship data. Ecology. **59**(1), 175–179 (1978) http://www.jstor.org/stable/1936645
7. Shanno, D.: On Broyden-Fletcher-Goldfarb-Shanno method. J. Optimiz. Theory. App. **46**(1), 87–94 (1985)
8. Bottou, L.: Large-scale machine learning with stochastic gradient descent. In: Lechevallier, Y., Saporta, G. (eds.) Proceedings of COMPSTAT'2010, pp. 177–186. Physica-Verlag HD, Heidelberg (2010)
9. Hager, W., Zhang, H.: A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM. J. Optimiz. **16**(1), 170–192 (2005). https://doi.org/10.1137/030601880
10. Celma, O., Herrera, P.: A new approach to evaluating novel recommendations. In: Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys '08, p. 179C186. Association for Computing Machinery, New York (2008). https://doi.org/10.1145/1454008.1454038
11. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inform. Syst. **22**(1), 5–53 (2004)
12. Nie, G., Zhang, L., Zhang, Y., Deng, W., Shi, Y.: Find intelligent knowledge by second-order mining: three cases from China. In: 2010 IEEE International Conference on Data Mining Workshops, pp. 1189–1195 (2010). https://doi.org/10.1109/ICDMW.2010.115
13. Zhang, L., Li, J., Li, A., Zhang, P., Nie, G., Shi, Y.: A new research field: intelligent knowledge management. In: 2009 International Conference on Business Intelligence and Financial Engineering, pp. 450–454 (2009). https://doi.org/10.1109/BIFE.2009.108
14. Wang, J., Zhang, Y.: Opportunity model for e-commerce recommendation: right product; right time. In: Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '13, p. 303C312. Association for Computing Machinery, New York (2013). https://doi.org/10.1145/2484028.2484067
15. Nash, J.C., Varadhan, R., Grothendieck, G.: ""Package optimr"". https://CRAN.R-project.org/package=optimr (2016)
16. Zhang, D., Shi, Y., Tian, Y., Zhu, M.: A class of classification and regression methods by multi-objective programming. Front. Comput. Sci. China. **3**(2), 192–204 (2009)
17. Zhao, X., Deng, W., Shi, Y.: Feature selection with attributes clustering by maximal information coefficient. Proc. Comput. Sci. **17**(1), 70–79 (2013)
18. Zhao, X., Shi, Y., Niu, L.: Kernel based simple regularized multiple criteria linear program for binary classification and regression. Intell. Data Anal. **19**(3), 505–527 (2015). https://doi.org/10.3233/IDA-150729
19. Richardson, M., Dominowska, E., Ragno, R.: Predicting clicks: estimating the click-through rate for new ads. In: WWW 2007 International World Wide Web Conference (2007)
20. Cheng, H., Cantu-Paz, E.: Personalized click prediction in sponsored search. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining, WSDM '10, p. 351C360. Association for Computing Machinery, New York (2010). https://doi.org/10.1145/1718487.1718531

21. Li, J., Zhang, P., Cao, Y., Liu, P., Guo, L.: Efficient behavior targeting using svm ensemble indexing. In: Proceedings of the 2012 IEEE 12th International Conference on Data Mining, ICDM '12, p. 409C418. IEEE Computer Society, New York (2012). https://doi.org/10.1109/ICDM.2012.152
22. Guo, Q., Agichtein, E.: Ready to buy or just browsing? Detecting web searcher goals from interaction data. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '10, p. 130C137. Association for Computing Machinery, New York (2010). https://doi.org/10.1145/1835449.1835473
23. Reshef, D.N., Reshef, Y.A., Finucane, H.K., Grossman, S.R., Mcvean, G., Turnbaugh, P.J., Lander, E.S., Mitzenmacher, M., Sabeti, P.C.: Detecting novel associations in large data sets. Science. **334**(6062), 1518–1524 (2011)
24. David, R., Yakir, R.: MINE software package. http://www.exploredata.net/
25. Reshef, D.N., Reshef, Y.A.: Supporting Online Material | Science. https://science.sciencemag.org/content/suppl/2011/12/14/334.6062.1518.DC1
26. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science. **315**(5814), 972–976 (2007). https://doi.org/10.1126/science.1136800
27. Shi, Y., Tian, Y., Kou, G., Peng, Y., Li, J.: Optimization Based Data Mining: Theory and Applications. Springer, New York (2011). https://doi.org/10.1007/978-0-85729-504-0
28. Vapnik, V.N.: The nature of statistical learning theory (1995)
29. Shi, Y.: Multiple criteria and multiple constraint levels linear programming: concepts, techniques and applications (2015)
30. Jayadeva, Khemchandani, R., Chandra, S.: Twin support vector machines for pattern classification. IEEE Trans. Pattern Anal. Mach. Intell. **29**(5), 905C910 (2007). https://doi.org/10.1109/TPAMI.2007.1068
31. Tian, Y., Qi, Z., Ju, X., Shi, Y., Liu, X.: Nonparallel support vector machines for pattern classification. IEEE Trans. Syst. Man. Cyb. **44**(7), 1067–1079 (2014). https://doi.org/10.1109/TCYB.2013.2279167
32. Watts, D., Strogatz, S.: Collective dynamics of 'small-world' networks. Nature. **393**(6684), 440–442 (1998). https://doi.org/10.1038/30918
33. Huang, Z., Ma, C., Xu, J., Huang, J.: Link prediction based on clustering coefficient. Appl. Phys. **4**, 101–106 (2014)
34. Geisser, S.: Predictive Inference: An Introduction. Chapman Hall, London (1993)
35. Lü, L., Zhou, T.: Link prediction in complex networks: a survey. Physica A. **390**(6), 1150–1170 (2011). https://doi.org/10.1016/j.physa.2010.11.027. https://www.sciencedirect.com/science/article/pii/S037843711000991X
36. Liu, Z., Zhang, Q.M., Lü, L., Zhou, T.: Link prediction in complex networks: a local naive Bayes model. Europhys. Lett. **96**(4), 48007 (2011). https://doi.org/10.1209/0295-5075/96/48007
37. Dong, Y., Ke, Q., Wu, B.: Link prediction based on node similarity. Comput. Sci. **38**(7), 162–164, 199 (2011) CSCD:4281527
38. Zhou, T., Lu, L., Zhang, Y.C.: Predicting missing links via local information. Eur. Phys. J. B. **71**(4), 623–630 (2009). https://doi.org/10.1140/epjb/e2009-00335-8
39. Liben-Nowell, D., Kleinberg, J.: The link prediction problem for social networks. In: Proceedings of the Twelfth International Conference on Information and Knowledge Management, CIKM '03, p. 556C559. Association for Computing Machinery, New York (2003). https://doi.org/10.1145/956863.956972
40. West, D.: Introduction to Graph Theory, 2nd edn. Prentice Hall, Hoboken, NJ (2001)
41. Erdős, P., Rényi, A.: On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci. **5**, 17–61 (1960)
42. Grossman, J.W.: Reviews: Small worlds: the dynamics of networks between order and randomness. Phys. Today. **31**(4), 74–75 (2002)

43. Marques Leite dos Santos, V., Brady Moreira, F., Longo, R.L.: Topology of the hydrogen bond networks in liquid water at room and supercritical conditions: a small-world structure. Chem. Phys. Lett. **390**(1), 157–161 (2004). https://doi.org/10.1016/j.cplett.2004.04.016. https://www.sciencedirect.com/science/article/pii/S0009261404005469
44. Barabasi, A., Albert, R.: Emergence of scaling in random networks. Science. **286**(5439), 509–512 (1999). https://doi.org/10.1126/science.286.5439.509
45. Albert, R., Barabasi, A.: Statistical mechanics of complex networks. Rev. Mod. Phys. **74**(1), 47–97 (2002). https://doi.org/10.1103/RevModPhys.74.47
46. Katzav, E., Nitzan, M., Ben-Avraham, D., Krapivsky, P.L., Khn, R., Ross, N., Biham, O.: Analytical results for the distribution of shortest path lengths in random networks. Europhys. Lett. **111**(2), 26006 (2015)
47. Newman, M.E.J., Watts, D.J.: Scaling and percolation in the small-world network model. Phys. Rev. E Stat. Phys. Plasmas Fluids Related Interdisc. Top. **60**(6 Pt B), 7332–7342 (1999)
48. Barrat, A., Weigt, M.: On the properties of small-world network models. Eur. Phys. J. B. **13**(3), 547–560 (2000)
49. Ventrella, A.V., Piro, G., Grieco, L.A.: On modeling shortest path length distribution in scale-free network topologies. IEEE Syst. J. **12**(4), 3869–3872 (2018). https://doi.org/10.1109/JSYST.2018.2823781
50. Cohen, R., Havlin, S.: Scale-free networks are ultrasmall. Phys. Rev. Lett. **90**(5), 058701 (2003)
51. Fang, W.: http://www.gd.chinanews.com/2018/2018-07-24/2/397998.shtml