A Markov Decision Process Social Recommender

Ruangroj Poonpol

SCPD HCP Student, 05366653

CS 299 Machine Learning Final Paper, Fall 2009

Abstract

In this paper, we explore the methodology to apply Markov Decision Process to the recommendation problem for the product category with high social network influence – in which the consumption decision is influenced by the user's social network. We chose to build "Food Recommender" as food choice is normally influenced by consumer's social network and food consumption (i.e. lunch, dinner) is often considered social process. We leverage the social interaction data from online social network of the author in order to obtain necessary social interaction information to calculate "social distance" feed into the model. We decided to apply the Markov Decision Process which treat recommendation process as sequential optimization problem rather than treating it as static prediction problem. We also apply other machine learning techniques such as collaborative filtering, knowledge-based and content-based recommender system to aid in the dictionary building and model parameters initialization which are crucial to the performance of the recommender. Over time, the result is slightly better than normal collaborative filtering technique and content-based recommender which utilizes simple similarity scoring model despite poorer initial result. However, when we explicitly display the "presence" of the persons in the user's social network that influences that particular recommendation the most, the performance of the recommender improved quite substantially. For example: rather than plainly recommending "Menu A", the recommender stated that " Menu A: your friend 's X liked it". Algorithmically speaking, the user friend's X saw this recommendation earlier and liked it, X 's pattern is similar to user's patter, and X is close to the user as measured by "social distance".

Research Process/Key Assumptions:

To build up the initial database, we conducted the survey of 100 users in the same online social network for top-of-mind most favorite foods which each user must submit at least top five choices. There are quite a few duplication so we have only 319 menus in total and the author used his domain knowledge to manually add another 181 ‘similar’ items so there are 500 menus in total in the Food Data Bank. We then add features to each menu so each menu is an eight-dimensional vector. Food ethnicity, vegetarian, main ingredient (meat), main ingredient (vegetable), cooking method, spiciness, food category, and special dietary. We then calculate the categorical similarity score based mainly on Match & Mismatch (M&M) score system for each pair of menu to get the proxy of the similarity of each pair of menus. This is essentially one form of clustering. The score is then adjusted manually using domain knowledge of the author. This Food Databank and similarity score will be used in a) initialization of the recommender parameters and b) to build the comparison model.

Then, we need to calculate the social distance among 100 users. To simplify the model, we use the average of the past 15 days of the interaction on online social network between each user pair and count each interaction as one despite its depth (length of post, number of response in the same thread, etc.). We also take into account the direction of interaction (i.e. who post on whose wall). Another simplification we apply to the model is that we assume zero interaction (and hence, arbitrarily large social distance) if a particular pair of users aren’t direct friends of each other on the online social network (i.e. degree of separation is > 1).

The Fundamental of the Model:

Markov Decision Process is a basically a four tuple: < S, A, Rwd, Tr>, where S is state, A is action, Rwd is the reward function that assigns a real value to each state-action pair, and Tr is the state-transition function which is the probability of the transition given action-state pairs. The goal of the agent is to maximize the sum of discounted reward and the optimal policy denoted π^* (s) is the policy that maximize sum of discounted reward. We will use the policy iteration to compute the optimal policy. To find the π^* and V^*, we search the space of possible policy by beginning with initial policy $\pi_0 (s) = \text{argmax}_{a \in A} [Rwd(s,a)]$. At each step, we compute the value function based on prior policy and update the policy given new value function:

$V_i (s) = Rwd (s, \pi_i(s)) + \gamma \sum_{sj \in S} tr (s, , \pi_i(s), s_i) V_i (s_j)$

$\pi_{i+1}(s) = \text{argmax}_{a \in A} [Rwd (s, a) + \gamma \sum_{sj \in S} tr (s, a, s_i) V_i (s_j)]$

We did conduct the experiment on users to calculate the initial model parameters especially transition function (Tr) :

We use simple maximum likelihood method to calculate the transition function from state s to s', we will use n-gram model which based on our design choice of top 5 most favorite to be 5-gram models meaning the probability that the user will click 'like' our recommendation is based on last one, two, three, four, five items he "like" earlier.

Hence, for example, the probability that users will like food item Fk after liking <F1,F2,F3,F4,F5> is:

Tr (< F1,F2,F3,F4,F5>, < <F2,F3,F4,F5,Fk>) =

count (<F1,F2,F3,F4,F5,Fk>)/count(<F1,F2,F3,F4,F5>)

We will further improve the model by: a) enhanced by some form of skipping (i.e. if a person 'liked' F1,F2,F3,F4,F5 then it is highly likely that some persons will like F5, after having chosen F1, F2, F3 too. Additionally, b) we will incorporate the clustering of food menu and social distance into the clustering of the state. For example, if food menus F1 , F2, F3, F4, F5 have very high similarity score to Fk then Fk+1, so the actual count of the occurrence of transition state from <F1,F2,F3,F4,F5> to Fk has higher weight than <F1,F2,F3,F4,F5> to Fk+1. We use sum of total square of similarity score as the measure of the "distance" from new food item to vector of prior state. Next, c) if we observe state transition from <F1,F2,F3,F4,F5> to Fk from users that are closer as measured by "social distance" , then that state transition receive higher weight. In essence, we measure similarity of state (s1, s2) as followed: Similarity(s1,s2) = $\sum \sum k1*k2 \ \delta (s1,s2)*(i+1)$

where δ (s1,s2) is the Kronecker delta function and k1 and k2 is the adjustment factor calculated from similarity of food menus and user social distance as mentioned above.

Lastly, rather than looking at “fixed” number of previous items (in this case five “prior” items) to calculate the most probable item to recommend, we will also consider predicting the next recommendation based on prior one, two, three, four items too and mix the model with the default five prior items that users “like”.

Note that, we use epsilon of 10% as the trade off constant between exploration and exploitation, so only +/- 10% of the optimal solution will be considered. Note that we use Boltzmann distribution to calculate this cut-off when we consider the ‘action’ that yielded the value function in that range.

Key Metrics:

For each session of recommendation (each recommendation) to test users (test size N= 35 out of total 100 users that we used to build the initial model parameters), we will calculate the percent “like” as the key measurement of the recommendation system. For example, if 18 out of 35 users “like” our recommendation, the performance is 51%.

We need to define the reward function, for our case, we have defined 3 reward values: +1 if user “like” the item, 0 if user is “neutral”, and -1 if user disliked the recommendation.

Comparison Model:

We used the standard Collaborative Filtering and Similarity Clustering model (based on maximum likelihood of the next item given previous sequence) to compare with our model.

Result:

Average Recommendation Performance Score	Collaborative Filtering	MDP Social Recommender
10	46%	39%
20	48%	43%
30	53%	47%
40	49%	54%

50	55%	59%

Initial results of MDP Social recommender is lower than collaborative filtering yet after 50 iterations, the result improved steadily and reached 59% which is slightly higher than Collaborative Filtering.

Model Improvement:

The result mentioned above from MDP Social recommendation doesn't provide significant improvement over standard CF model and hence doesn't justify substantial model complexity and computation headcount; so we come up with improvement idea whereby we explicitly mention the name of the person that most influence model outcome together with the recommendation of the menu and after 50 iterations, we yielded 69% which is +14% above CF.

Concerns and Conclusion:

Based on the above result, one key hypothesis we have is that the social dimension of recommendation especially in relatively small data environment like this setting, doesn't make Markov Decision Process to be obviously superior to standard simpler method like Collaborative Filtering. The substantial improvement will happen only if we make the social element "explicit" in the recommendation by simply stating which node in the social network that resulted in the recommendation. However, we don't know if the improvement in recommendation performance is due to the algorithm or simply from the appearance of the 'influencer' in the social network in the recommendation which essentially might change the initial decision of the users. In essence, what we are building might not be recommendation system but 'influencing system'. In addition, this definitely will cause privacy concerns to users whose name is used to recommend products which in practice could be serious problem. Lastly, we should also consider exploring further on the direction of the social interaction and the structure of the network on the recommendation process. For example, will the influence of the users that have high connection and act as the hub be higher than the users that proactively comment/post/respond o other people's wall