A Simple Matching Model

Jasprit Singh Aujla

December 5, 2023

0.1 Introduction

The purpose of this short paper is to introduce a simple framework to solve two-sided matching problems. This framework has potential applications to problems in computational advertising - matching ads to consumers and in social network settings - e.g., matching potential friends/connections to users or matching job candidates to recruiters on LinkedIn.

In computational advertising, advertisers face the problem of choosing which ads to show to which consumers. This can be viewed as a problem of finding the best match between the ad and the context. In this case, the context refers primarily to the consumer, but it also may include the time of day and the website on which the advertiser is seeking to purchase ad space.

Similarly, social media platforms face a matching problem when they attempt to suggest potential new friends or connections to a user. Users are more likely to be a friend or connection with someone with whom they share mutual friends or connections and whose interests are aligned with their own. LinkedIn, the employment-focused social media platform, also tries to match potential job candidates to recruiters.

The problems described above may be viewed as two-sided matching problems. A consumer clicking an ad or a social media user sending a connection request (and the receiver accepting) is essentially a result of the interaction of both sides solving their own optimization problems. The ad agency displays its ad to that consumer because it believes that the consumer has a high chance of clicking on the ad, and the consumer clicks on the ad because he or she is interested in the ad. Similarly, the social media user who sent the connection request likes or is interested in the user he or she is sending the request to and the receiver is indicating his or her interest by accepting.

Our aim is to demonstrate the feasibility of our simple matching model which can be applied to match consumers to ads, social media users to new friends and recruiters to job applicants. We do this by sketching out a hypothetical example involving simulation data where an advertising agency chooses which ads and consumers to purchase ad space for. We also test our model's performance in predicting connections on a social media network using data from Facebook.

0.2 Literature Review

LinkedIn Recruiter Recommendation System:

A paper by researchers at LinkedIn (Guo et. al 2019), is perhaps most closely related to our proposed framework. This paper outlines the details behind LinkedIn's Recruiter product - a recruiter search and

recommendation system. The system provides a ranked list of candidates corresponding to a search request in the form of a query, a job posting or a recommended candidate. Given a search request, candidates that match the request are selected and then ranked based on a variety of factors (such as the similarity of their work experience/skills with the search criteria, job posting location and the likelihood of a response from an interested candidate) using machine-learned models in multiple passes.

The aim of the system built by the researchers was to enable recruiters to identify "talent pools" that were optimized for the likelihood of making a successful hire. A key challenge for the algorithm was that it required mutual interest between the recruiter and the candidate i.e., it was necessary for the candidate contacted by the recruiter to show interest in the job opportunity in order for there to be a successful match.

Their first machine learning model was a linear model before they eventually employed Gradient-Boosted Decision Trees (GBDT). GBDT had a few key advantages including being able to work well with feature collinearity, handling features with different ranges and missing feature values.

Next, I outline two general frameworks - Discrete Choice models and the Gale-Shapley algorithm - which are similar in spirit to the type of matching models that we are trying to build.

Discrete Choice (Random Utility Model):

Suppose a decision maker i faces a finite set Ω_J of dimension J.

The utility of choice j for individual i can be written as:

$$U_{ij} = V_{ij} + \epsilon_{ij}$$

j is chosen in Ω_J if $\quad U_{ij} - U_{ij'} > 0 \; \forall \; j' \neq j \in \Omega$

$V_{ij} = V(\mathbf{x_i}, \mathbf{v_j}, \theta)$ is a linear index which contains:

1. Alternative specific attributes (in $\mathbf{v_j}$), a $L \times 1$ vector of attributes for the jth alternative
2. Individual characteristics (in $\mathbf{x_i}$), a $K \times 1$ vector of alternative-invariant characteristics
3. Interaction of attributes and individual characteristics
4. θ - a vector of unknown parameters

The randomness in the random utility model arises due to the ϵ_{ij} term which can be viewed as unobserved heterogeneity in tastes for the product or for attributes of the product. Each decision maker i has an ϵ_{ij} value for every one of the alternatives.

Assume that $\epsilon_{ij} \overset{\text{iid}}{\sim}$ Type I Extreme Value,

$$\begin{aligned} P_{ij} &= Pr(U_{ij} > U_{ik}) && (1) \\ &= Pr(V_{ij} + \epsilon_{ij} > V_{ik} + \epsilon_{ik}) && (2) \\ &= Pr(\epsilon_{ik} < \epsilon_{ij} + V_{ij} - V_{ik}) && (3) \end{aligned}$$

Suppose we know ϵ_{ij}, V_{ij} and V_{ik}. For a single ϵ_{ik}, this probability is the cumulative distribution of a Type I Extreme Value random variable.

$$Pr(\epsilon_{ik} < \epsilon_{ij} + V_{ij} - V_{ik} | \epsilon_{ij}) = e^{-e^{-(\epsilon_{ij} + V_{ij} - V_{ik})}}$$

This is the probability for a single alternative k. We would like to know this probability for all $k \neq j$. Since ϵ_{ik} is i.i.d., we can take the product of the probability for each ϵ_{ik}:

$$P_{ij} | \epsilon_{ij} = \prod_{j \neq k} e^{-e^{-(\epsilon_{ij} + V_{ij} - V_{ik})}}$$

Since ϵ_{ij} is random, we integrate over the density of ϵ_{ij}.

$$P_{ij} = \int \left(\prod_{j' \neq j} e^{-e^{-(\epsilon_{ij} + V_{ij} - V_{ij'})}} \right) e^{-\epsilon_{ij}} e^{-e^{-\epsilon_{ij}}} d\epsilon_{ij} \tag{4}$$

$$= \frac{e^{V_{ij}}}{\sum_{k=1}^{J} e^{V_{ik}}} \tag{5}$$

Therefore, the probability that decision maker i chooses alternative j depends on the representative utility of all the other alternatives $k \neq j$.

Some key properties of the choice probabilities P_{ij} are that they sum to 1 and they are a sigmoidal function of representative utility and so marginal effects are small when probabilities are close to 0 and marginal effects are largest when the probability is equal to 0.5.

If we assume representative utility is linear in parameters $V_{ij} = \beta' x_{ij}$ then the logit choice probability has the simple closed form:

$$P_{ij} = \frac{e^{\beta' x_{ij}}}{\sum_k e^{\beta' x_{ik}}}$$

In the case of a binary logit model, we can estimate this model in R and calculate log odds ratios. This model can also be extended to capture multiple alternatives. If we group alternatives into nests to allow for correlations between unobserved components of utility, we get the nested logit model.

An even more flexible version of this is the mixed logit model. This model allows for a richer representation of preference variation, more flexible substitution patterns and allows for the use of longitudinal data. The mixed logit model achieves this flexibility by assuming that there is a distribution of β coefficients in the population.

A mixed logit model is any discrete choice model with choice probabilities of the form:

$$P_{ij} = \int L_{ij}(\beta) f(\beta | \theta) d\beta$$

where $L_{ij}(\beta)$ is the logit probabilities at a given set of coefficients β

$$L_{ij}(\beta) = \frac{e^{V_{ij}(\beta)}}{\sum_{k=1}^{K} e^{V_{ik}(\beta)}}$$

and $f(\beta|\theta)$ is a density function of coefficients β which depend on a vector of parameters θ. The idea is that the logit probabilities are evaluated at different values of β and each logit choice probability is weighted by the density $f(\beta|\theta)$. Thus, the mixed logit choice probability is a mixed function of logit choice probabilities $L_{ij}(\beta)$ with the mixing distribution $f(\beta|\theta)$. A weighted average of logit choice probabilities evaluated at different values of β in the population. Weighting by the density of the β's i.e., how likely the β's are in our population.

Mixed logit choice probabilities are not closed-form and estimation has to be done via numerical simulation and θ is the object of interest. The β's can then be recovered from the estimated θ.

Mixed Logit Model Specification:

$$U_{ij} = \beta_{\mathbf{i}}' \mathbf{x_{ij}} + \epsilon_{ij}$$

where x_{ij} is the data for alternative j and decision maker i, β_i is an individual-specific vector of coefficients and ϵ_{ij} is an i.i.d. extreme value term.

$$L_{ij}(\beta_i) = \frac{e^{\beta_{\mathbf{i}}' \mathbf{x_{ij}}}}{\sum_{k=1}^{K} e^{\beta_{\mathbf{i}}' \mathbf{x_{ij}}}}$$

This quantity has to be integrated with respect to the distribution of the betas to get the unconditional choice probability, thereby resulting in the equation above.

A discrete choice model is useful in telling us how to choose one alternative from among a set of different alternatives so it's useful to model how consumers choose between different cars for example with different characteristics.

Gale-Shapley algotrithm:

A second framework that is relevant when considering matching-type problems is the Gale-Shapley algorithm (1962). The algorithm was conceived to solve the stable matching problem where a proposer seeks to arrange marriages between an equal number of males and females. A marriage is stable as long as there was no instance where two people simultaneously prefer each other to their assigned spouses.

The algorithm goes like this: Each man and each woman ranks the members (from 1 to n) of the opposite sex using a preference list. On the first day each woman proposes to her number one choice. Some of the men receive multiple proposals, while others receive none. The men who receive multiple proposals will reject all proposals they receive except the proposal from the one who is ranked most highly by them. At the end of this round, some of the women have made tentative proposals that have not been rejected yet. These couples are said to be tentatively engaged. On day 2, each woman whose proposal was rejected the day before proposes to her next best choice, regardless of whether that man is tentatively engaged or not. Some men receive new proposals while others do not receive any. Therefore, some men are now able to trade up and break their previous engagement if the new proposal is preferred to their current partner.

This process is repeated on day 3, 4 and 5. Any woman who is rejected the day before proposes to her next choice and men again have the chance to trade up if they receive better proposals. At some point, this stops.

The algorithm leads to the following nice results:

- Theorem 1: The algorithm arranges stable marriages in the sense outline above.

- Theorem 2: The algorithm assigns every woman simultaneously her best possible husband.
- Theorem 3: The algorithm assigns every man their worst possible wife among all sets of possible stable marriages.
- Theorem 4: It is not possible to game this system. There is no advantage to lying at all.

Gale and Shapley's algorithm has been applied to real-world matching problems. For example, it has been used to match graduating medical school students to hospitals. All graduating medical students submit preference lists of residencies and hospitals submit preference lists of the students. A computer then uses exactly this algorithm to assign the graduating medical students to residency programs.

0.3 Our Model

This section contains details about our model in the context of online advertising. The setting is as follows: Imagine you run an advertising agency and have access to 100 ads that you can choose to buy ad space for. You also have access to information about 1000 users.

Our goal is to maximize the match between a consumer who is viewing the ad and an advertiser who is bidding for the same ad. We can define the following quantities:

$$p_{ij} = Pr[Consumer\ i\ is\ interested\ in\ ad\ j]$$
$$q_{ji} = Pr[Company\ bidding\ on\ ad\ j\ is\ interested\ in\ consumer\ i]$$

We should match consumer i to ad j if the product of these quantities $p_{ij} \times q_{ij}$ is large.

This problem can be broken down into two smaller problems:

1. The probability p_{ij} of consumer i being interested in a given ad j
2. The probability q_{ji} of an advertiser for ad j being interested in consumer i

It is convenient to model $logit(p_{ij}) = log(\frac{p_{ij}}{1-p_{ij}}) = \beta_0 + \Sigma\beta_i x_i$. Similarly $logit(q_{ji}) = log(\frac{q_{ji}}{1-q_{ji}}) = \gamma_0 + \Sigma\gamma_i z_i$ where x_i are the covariates which influence the probability of a consumer being interested in a given ad and z_i are the covariates which influence the probability that a company bidding on the ad has interest in a given consumer.

The probability of a consumer being interested in a given ad depends on (among other things):

- demographic factors including age, sex, etc.
- the time of day
- their tastes and preferences

On the other hand, the probability of a company bidding on the ad being interested in a given consumer depends on:

- demographic factors including age, sex, etc. which are observable to the company bidding on the ad (through the data that the company has on users). For example, a pizza chain's typical consumer might be male, and aged between 18-25.

- the time of day (continuing with the pizza example, the company would be more interested in showing the consumer an ad close to meal times). Other contextual information may also be relevant (e.g., pizza ads may spike during the day of the Super Bowl)
- how well the consumer's tastes and preferences align with what the company is trying to sell in the ad. The company might have access to the consumer's search history and based on that, can make deductions about what the consumer is interested in.
- the company's advertising budget. If the company has a large budget, it may be more willing to bid for an ad even though the consumer's demographics and preferences are not as well-aligned with the company's target groups.
- the company has a click-through-rate (CTR) target. If the company wants to achieve a certain CTR, it may be more selective in choosing to bid on ads so that it only shows ads to consumers who it believes have a higher chance of clicking on the ad.

0.4 A Toy Example - Tesla Ads

In this section, we use a toy example to fix ideas. Consider the problem of an advertising agency choosing whether to serve advertisements about Tesla vehicles to users online. Assume that the agency has some information about users' behaviour online (including their online activity score which we represent using a scale between 1 and 10), and has some idea about their income (through the information it has on the average income in that area).

The following are the covariates for the advertiser's problem and the consumer's problem.

Advertiser's problem covariates:

- Income (in tens of thousands)
- Age
- Gender (0 for male, 1 for female)
- Whether the consumer is a car owner (0,1)
- The consumer's online activity score (1-10)
- Whether the customer is environmentally conscious (0,1)

Consumer's problem covariates:

- Income (in tens of thousands)
- Age
- Gender (0 for male, 1 for female)
- Whether the consumer is a car owner (0,1)
- Whether the consumer has a friend or family member who owns a Tesla (0,1)
- Whether the consumer lives close to a Tesla dealership (0,1)
- Whether the consumer is environmentally conscious (0,1)

- Whether the consumer works in the tech industry (0,1)

We can simulate data from each model by choosing reasonable values for each of the coefficients of the covariates, calculating the implied probabilities (by taking the expit function) and drawing from a Bernoulli distribution with that probability. This returns a 0 or a 1 for the outcome variable.

Next, using the simulated data, we fit two logistic regression models and attempt to recover the coefficients. Doing this on a sample size of 1000 observations we get an Area Under Curve (AUC) score of 0.88 for the consumer's problem and an AUC score of 0.95 for the advertiser's problem.

Tables 1 and 2 shows the actual coefficient and the predicted coefficients for the two problems.

	Actual Coefficient	Fitted Coefficient	Fitted Std. Error	Fitted Pr($>\|z\|$)
Intercept	-6.0000	-6.0062	0.6628	<2e-16***
Income	0.2000	0.1903	0.0134	<2e-16***
Age	-0.2000	-0.1808	0.01263	<2e-16***
Gender	0.3000	0.5880	0.2141	0.0060**
Car Owner	0.0500	-0.0029	0.2110	0.9889
Online Activity	0.2000	0.1657	0.0351	2.36e-06***
Environmentally Conscious	0.6000	0.5364	0.2128	0.0117

Table 1: Advertiser Problem Results

	Actual Coefficient	Fitted Coefficient	Fitted Std. Error	Fitted Pr($>\|z\|$)
Intercept	-5.0000	-4.6734	0.6001	6.84e-15 ***
Income	0.1000	0.0961	0.0082	<2e-16 ***
Age	-0.1000	-0.1060	0.0090	<2e-16 ***
Gender	0.2000	0.4445	0.1944	0.0222 *
Car Owner	0.0500	-0.1455	0.1932	0.4516
Has Tesla Connection	0.7000	0.5496	0.1956	0.0050 **
Near Dealership	0.4000	0.2825	0.1935	0.1442
Environmentally Conscious	0.5000	0.6655	0.1962	0.0007 ***
Tech Occupation	0.4000	0.6194	0.1957	0.0015 **

Table 2: Consumer Problem Results

Finally, we multiply the two probabilities together to get a (1000,1) vector where each entry reflects the strength of the match between each consumer and the Tesla ad.

The data presented in this simulation is sufficient only to fit a model for one particular ad (for Tesla vehicles). To operationalize our idea in a real-world setting, we would need to consider multiple ads that the agency could show at any one time. For each of these ads, we would need to compute the probability that a given consumer is interested in that ad.

Suppose again that you run an advertising agency and have access to 100 ads that you can choose to buy ad space for. You also have access to information about 1000 users (including the covariates specified above). In order to find the probability that a given user, i likes a given ad j, we fit a logistic regression for that user and estimate p_{ij}. We do the same for the rest of the 999 users, to find the probability that they like that ad

too. Then, turning to the advertiser's side of the problem, we want to estimate the probability that the ad j "likes" the consumer i. To do this, we fit a logistic regression for ad j and estimate q_{ji}. This is repeated for all other 99 ads. Then we compute the pairwise probability of every user i with every ad j. Out of this whole universe, we might choose to buy ad space for the top 100 (consumer,ad) pairs that have the highest product of probabilities.

0.5 Application - Facebook Friend Recommender System

In this section, we apply our model to build a friend recommender system and test its performance using data from Facebook.

0.5.1 Data

The data set we base our analysis on was first used in a paper by Traud et. al (2011). This paper studied the social structure of Facebook's friendship networks at one hundred American colleges and universities at a single point in time in September 2005. Facebook users at the time were only able to connect with other users from the same institution, hence all connections between individuals were within the same institution. As a result, we consider data from a single institution - Duke - for our analysis.

The data was split into two data frames. In the first data frame, each row represented an individual and contained the following information:

- student/faculty status flag
- gender
- major
- second major/minor (if applicable)
- residence (dorm/house)
- year
- high school

High school, major and residence were represented using anonymous numerical identifiers. All covariates were categorical and missing data was coded as 0.

The second data frame consisted of a sparse matrix which encoded the connections between all individuals e.g., the vector (1,4056) would indicate that there was a connection between individual 1 and individual 4056.

To prepare the data for analysis, it was necessary to create a new column in the data frame where each entry contained a list of all connections for that individual. In addition, the following steps were taken to process the data:

1. Second major/minor was dropped as a covariate as it had many missing values and was unlikely to be a strong predictor for the probability of a given individual liking another individual
2. All observations with at least one missing value were dropped
3. All observations with fewer than 50 connections were dropped

The final data set consisted of 3,247 observations.

0.5.2 Methods

Just as in the Tesla ads example, we can break this problem into two pieces. The first is the probability that individual i likes individual j. The model specification for this problem is:

$$log\left(\frac{Pr(y_{ij}=1)}{1-Pr(y_{ij}=1)}\right) = \beta_0 + \sum_s \beta_{1,s} I\left(\text{Status}_j = s\right) + \sum_g \beta_{2,g} I\left(\text{Gender}_j = g\right) + \tag{6}$$

$$\sum_m \beta_{3,m} I\left(\text{Major}_j = m\right) + \sum_r \beta_{4,r} I\left(\text{Residence}_j = r\right) + \tag{7}$$

$$\sum_y \beta_{5,y} I\left(\text{Year}_j = y\right) + \sum_h \beta_{5,h} I\left(\text{High School}_j = h\right) \tag{8}$$

where y_{ij} is the probability that individual i likes individual j.

Similarly, the model specification for individual j is

$$log\left(\frac{Pr(y_{ji}=1)}{1-Pr(y_{ji}=1)}\right) = \beta_0 + \sum_s \beta_{1,s} I\left(\text{Status}_i = s\right) + \sum_g \beta_{2,g} I\left(\text{Gender}_i = g\right) + \tag{9}$$

$$\sum_m \beta_{3,m} I\left(\text{Major}_i = m\right) + \sum_r \beta_{4,r} I\left(\text{Residence}_i = r\right) + \tag{10}$$

$$\sum_y \beta_{5,y} I\left(\text{Year}_i = y\right) + \sum_h \beta_{5,h} I\left(\text{High School}_i = h\right) \tag{11}$$

where y_{ji} is the probability that individual j likes individual i.

Since the problems for both individual i and individual j are symmetric i.e., they involve the same covariates, we could fit the same model (using slightly different data) for every individual.

Due to the large number of categories (for example there were over 80 different high schools alone), it was not feasible to run a standard logistic regression since there were many covariates relative to the number of data points. Therefore, we ran a regularized (LASSO) logistic regression using the **glmnet** package in R. λ was chosen to minimize the mean cross validated error.

Doing this we obtained 3,247 logistic models in total. These were used to calculate the probability vector for a given individual liking every other individual. This was then repeated for every individual, giving a total of 3,247 probability vectors, each with a length of 3,246 (the probability of an individual liking himself/herself was ignored).

Next, we took the pairwise product between an individual's probability of liking another individual and that other individual's probability of liking the first individual. In other words we calculated $Pr(y_{ij} = 1) \times Pr(y_{ji} = 1)$ for a given individual. This was again repeated for all individuals, giving a matrix with 3,247 rows and 3,247 columns where the (i, j)th entry was the product $Pr(y_{ij} = 1) \times Pr(y_{ji} = 1)$.

$$\begin{pmatrix} NA & p_{1,2} * p_{2,1} & p_{1,3} * p_{3,1} & \dots & p_{1,3247} * p_{3247,1} \\ p_{1,2} * p_{2,1} & NA & p_{2,3} * p_{3,2} & \dots & p_{2,3247} * p_{3247,2} \\ p_{1,3} * p_{3,1} & p_{2,3} * p_{3,2} & NA & \dots & p_{3,3247} * p_{3247,3} \\ & \dots & \dots & & \\ p_{1,3247} * p_{3247,1} & p_{2,3247} * p_{3247,2} & p_{3,3247} * p_{3247,3} & \dots & NA \end{pmatrix}$$

$p_{ij} \times p_{ji}$ should be interpreted as being related to the probability of a match between individual i and individual j. However, its magnitude has no intrinsic meaning. It only serves to reflect how good of a match two individuals are for each other. Higher absolute values of this product indicate a better match. The entries on the diagonal of the matrix above were coded as NA since we are not interested in the probability of an individual being matched to himself or herself.

0.5.3 Results

To generate prediction of the class for each individual pair (1 for a connection, 0 for no connection), we choose a threshold of 0.008 in order to balance precision and recall (more on this below). If the product $p_{ij} \times p_{ji}$ is greater than this threshold then we predict that there is a connection between individual i and individual j, otherwise we predict that there is no connection.

We then compare these predictions with the actual values of whether there was a connection between two given individuals. The resulting confusion matrix, for the $\frac{(3{,}247 \times 3{,}246)}{2} = 5,269,881$ possible (unique) pairwise connections looks like this:

		Actual Connections		
		1	0	Total
Predictions	1	114,833	59,661	174,494
	0	48,094	5,047,293	5,095,387
	Total	162,927	5,106,954	5,269,881

There are several metrics that we could use to evaluate the performance of this matching model. TP refers to True Positives, TN refers to True Negatives, FP refers to False Positives and FN refers to False Negatives.

1. $$Accuracy = \frac{\#TP + \#TN}{\#TP + \#TN + \#FP + \#FN} = 0.980$$

2. $$Precision = \frac{\#TP}{\#TP + \#FP} = 0.658$$

3. $$Recall = \frac{\#TP}{\#TP + \#FN} = 0.705$$

4. $$F1\ Score = \frac{2 \times precision \times recall}{precision + recall} = 0.681$$

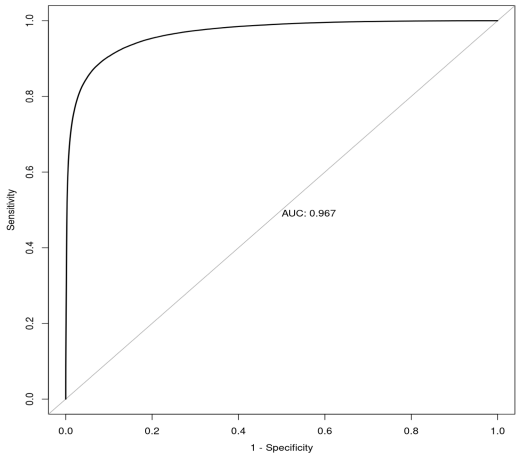

Figure 1: ROC Curve

This is a classic case of imbalanced data (there are many more 0s than 1s). Thus it made sense to use precision and recall as the appropriate metrics since accuracy will be very good if the model correctly predicting all 0s. The threshold was chosen to strike a balance between precision - which captures how many irrelevant recommendations we make - and recall - which captures how many of the total 1s we correctly predicted. A precision of 65.8 percent and 70.5 percent recall is decent performance. The F1 score - which is a harmonic mean of precision and recall - was also a respectable 0.681. The Receiver Operating Characteristic (ROC) curve is shown in Figure 1. The AUC is 0.967.

Given that we are building a recommender system, we might be interested in a metric that is able to capture how many of our first k recommendations are accurate. To do this, we first order our friend recommendations for each user in decreasing order of $p_{ij} \times p_{ji}$. Suppose say we choose k to be 10 so that we display the top 10 friend recommendations to each user.

We can then compute the precision at this cutoff of 10. Table 3 shows the top 10 recommendations that are made to individual 2 and whether or not each recommendation was an actual connection.

The precision at a cutoff of 10 is then computed as $P(k = 10) = \frac{9}{10}$ since we 9 out of our top 10 recommendations turned out to be actual connections. Note that a recommendation system would filter out the individuals which are already connected to individual 1 (i.e., have an Actual Connection = 1) and recommend the top 10 individuals who are not currently connections.

Going further, we can also define a quantity called the average precision. Suppose we are asked to recommend N potential connections, and the number of actual connections and we have information on whether or not these N users are current connections with the user we made the recommendation to. Then,

$$Average\ Precision(N) = \frac{1}{N} \sum_{k=1}^{N} P(k)$$

where $P(k)$ is the precision at k. The more correct recommendations the system gives, the larger the average

Rank	Individual Id	$p_{ij} \times p_{ji}$	Actual Connection
1	8869	0.529	1
2	6184	0.517	1
3	7343	0.444	1
4	2428	0.436	1
5	6581	0.435	1
6	4260	0.363	1
7	9505	0.303	1
8	3778	0.290	1
9	5462	0.229	0
10	1503	0.222	1

Table 3: Recommendations for Individual 1

precision. Also, the calculation of average precision rewards the system for front-loading the recommendations that are most likely to be correct since they get a higher weight. As an example, the average precision for individual 1 is computed as

$$\frac{1}{10}\left(\frac{1}{1}+\frac{2}{2}+...+\frac{8}{9}+\frac{9}{10}\right)=0.979$$

We can compute the average precision for the rest of the 3,246 users in a similar fashion. Averaging the average precision over all the users, we arrive at the Mean Average Precision or MAP. In this case, it is equal to 0.626.

0.5.4 Discussion

The model's performance on the full dataset is good. However, it should be emphasized that this is a relatively favorable example in that the covariates available - especially academic major and place of residence - are likely to be highly predictive of whether or not any two given people were friends on Facebook. Other real-world examples are likely to be less favorable. Nevertheless, it is promising that the model performs relatively well in this context.

0.5.5 Test Set

You may have noticed that we did not use a test set to evaluate the model's performance. Due to the two-sided nature of this problem, the test set has to be constructed in a non-standard way.

Here, we outline how this can be done. Suppose we split our data into a training and a test set. First, we train our model on the training data and retrieve the coefficient estimates for each individual liking every other individual in the training set. Then we would use these estimated coefficients to predict how much each individual in the training set likes every individual in the test set based on the test data. To calculate the pairwise probabilities for all individuals in the test set with all individuals in the training set, it is also necessary to find the probability that each individual in the test set likes each individual in the training set. To do this, we need to fit a model for each individual in the test set using data from the other individuals in

the test set. Then, we use these models to predict the probability of every individual in the test set liking every individual in the training set.

Since the test and training sets are essentially treated symmetrically, it makes sense to use a 50-50 split of the training and test sets. In this case that would mean about 1,633 observations in the training set and 1,633 observations in the test set. We would then calculate the pairwise probabilities between every individual in the training set and every individual in the test set. This would give us a total of $\frac{1632 \times 1633}{2} = 1,332,528$ non-duplicated pairwise probabilities. We can then set a threshold and evaluate the model's performance as we did for the whole data set above.

0.6 Conclusion

In conclusion, this paper has proposed a simple framework for two-sided matching problems. The feasibility and performance of this framework has been evaluated using an example where we try to recommend Facebook users whom we think will be a good match to one another. We also demonstrate how this framework can be applied to a computational advertising problem in a hypothetical setting where an advertising agency is choosing which ads and users it wants to purchase ad space for. A key advantage of our model is its simplicity and therefore its interpretability.

References

Qi, G., Geyik, S. G., Ozcaglar, C., Thakkar, K., Anjum, N., &; Kenthapadi, K. (n.d.). The AI behind LinkedIn Recruiter Search and recommendation systems. LinkedIn Engineering. https://engineering.linkedin.com/blog/2019/04/ai-behind-linkedin-recruiter-search-and-recommendation-systems

Gale, D., & Shapley, L. S. (1962). College Admissions and the Stability of Marriage. The American Mathematical Monthly, 69(1), 9–15. https://doi.org/10.2307/2312726

Amanda L. Traud, Peter J. Mucha, Mason A. Porter. Social structure of Facebook networks. Physica A: Statistical Mechanics and its Applications, Volume 391, Issue 16, 2012, Pages 4165-4180, ISSN 0378-4371, https://doi.org/10.1016/j.physa.2011.12.021.