MALMÖ UNIVERSITET

Faculty of Technology and Society
Department of Computer Science and
Media Technology

Bachelor Thesis
15 ECTS Credits, Undergraduate Level

Building a Recommender System for Speed Dating

Robin Sääf

Degree: Bachelor of Science 180 Credits
Major: Computer Science
Programme: System Developer
Date of Final Seminar: 2018–08–27

Supervisor: Dimitris Paraschakis
Examiner: Radu-Casian Mihailescu

Abstract

Dating is a popular activity worldwide for finding a potential partner. Users of matchmaking services explore different dating formats to establish the relationship they desire. While some prefer online services such as mobile dating apps, others prefer the experience of physical interaction in speed dating. To further aid visitors in the fast-paced speed dating environment, recommender systems can be used for assistance in mate selection.

This thesis proposes a method for increasing efficiency in speed dating, by detailing the fundamental principles of building a recommender system that suggests potentially interesting partners among speed dating attendees. Recommendations are based on the preferences and self-descriptions of speed dating attendees, and the problem that is addressed is how to build a classification model that is able to predict the degree of interest participants will have in each other. Two classifiers are trained and evaluated using cross validation on a real-world speed dating dataset, and compared using the F1 score and Precision-Recall AUC evaluation metrics.

The results that are achieved show that it is possible to develop a speed dating recommender system using fairly simple classification models. The proposed system can be seen as a minimum viable product, because the level of accuracy of the recommendations would likely need to be improved in order to be useful in a production environment.

Sammanfattning

För människor över hela världen är dating ett populärt tillvägagångssätt för att hitta partners. Användare av matchmaking-tjänster utforskar olika varianter av dating med målet att kunna upprätta ett lyckat förhållande. Medan vissa föredrar onlinetjänster såsom mobil-applikationer, föredrar andra att kunna träffas i verkligheten, som i speed dating. För att hjälpa besökare att navigera i den intensiva speed dating-miljön kan rekommendationssystem användas som ett sätt att underlätta valet av partner.

Den här uppsatsen föreslår en metod för att öka effektiviteten i speed dating, genom att beskriva de grundläggande byggstenar som behövs för att utveckla ett rekommendationssystem som är kapabelt att föreslå vilka speed dating-deltagare som kan vara av intresse för varandra. Rekommendationerna är baserade på deltagares preferenser och beskrivningar av sig själva, och uppgiften är att bygga en klassificeringsmodell som kan förutsäga sannolikheten att olika deltagare är intresserade av varandra. Tre olika klassificerare utvecklas och utvärderas med hjälp av korsvalidering mot ett riktigt speed dating-dataset, och jämförs genom att mäta deras F1-värde och yta under en Sensitivitet-Specificitet-kurva.

Resultaten som uppnås visar på att det är möjligt att utveckla ett rekommendationssystem för speed dating medelst förhållandevis enkla klassificeringsmodeller. Det föreslagna systemet kan ses som en konceptvalidering, eftersom rekommendationernas träffsäkerhet sannolikt måste förbättras för att kunna användas i en produktionsmiljö.

Contents

1 Introduction

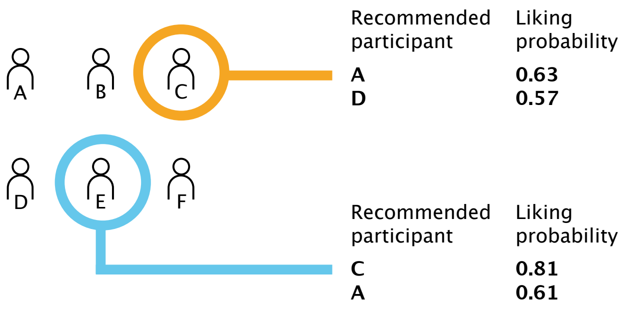
The process of finding a partner is prevalent in most people's life at some point, be it to find a short fling or a lifetime companion. Deciding who to spend time with is a task that requires a lot of conscientious thought and sometimes less informed trial-and-error. With the use of matchmaking services, people who share the quest for establishing a romantic relationship get exposed to each other by the means of online dating or real life arrangements such as speed dating.

In speed dating, participants are joined to have a short date which usually lasts 4-8 minutes with each participant of the opposite sex. In the end, each participant is asked to decide whether they would care for another date with their new acquaintance, receiving their partner's contact details in the case of a mutual positive outcome. The speed dating model is formed in a way that enables meeting a lot of people in a short period of time. Attendees are therefore likely to seek ways to increase the efficiency in their search of a partner.

The purpose of this thesis is to unfold the nature of building a recommender system for speed dating. A recommender system that is able to suggest useful recommendations could easily be of interest to someone who not only wants to find a partner quickly but also prefer some assistance in mate selection. For other users it might serve as an opportunity simply to get a feel for each other before even meeting. If anything it might serve as a conversation starter for participants to discuss whether they agree on the system's decision. This paper studies the following research question:

RQ: How to build a recommender system for a speed dating dataset?

The issue at hand is a classification problem. The recommender system uses the characteristics and preferences of one participant and their date to analyse an instance of a date, and does this for all the speed dates. For each participant, the system will find other participants that seem relevant, rank them in the order of estimated preference, and then recommend to the user. Recommendations are assumed to be given to participants who have just arrived to the event, meaning that their past dating history is unknown to the system, which is known as a cold-start scenario.



Recommended participant	Liking probability
A	0.63
D	0.57

Recommended participant	Liking probability
C	0.81
A	0.61

Figure 1: The visitors at a speed dating event receive a list of personalised recommendations including the estimated likelihood of affinity for each recommendation.

2 Background

This section begins with giving an overview of the matchmaking domain and how it in recent years has become entangled with machine learning systems. Thereafter, common applications for recommender systems are discussed as well as their typical technical representations.

2.1 The Matchmaking Domain

Dating is an activity that is employed by people all around the globe. The introduction of the Internet enabled a new world of possibilities for meeting relationship prospects through websites and mobile applications. From 2013 to 2015, a persistent rise was observed in the amount of people in the US in ages 35-64 who reported engaging in either online dating or dating-related mobile apps [1]. For people aged 18-24 the number was nearly tripled. Pioneering work from early 1990's [2] showed that people are able to successfully make accurate social judgements based on a few minutes of interaction. This indicates that the traditional model of all-night dating is obsolete for people who seek to find a partner quickly.

A standout factor when using the speed dating model is that it can function as a hybrid between traditional all-night dates and the popular model of online dating. Participants may enjoy the experience of having a real life meeting with a potential partner, also knowing that the encounter is short-lived, in case parties do not "click". For researchers, the opportunities of studying these numerous brief meetings are also considerable in terms of possible variation in experiments [3].

While recommender systems have been around since the mid 1990's, the research field of constructing systems for matchmaking is still fairly new. One of the earliest publications of recommender systems related to matchmaking is [4] from 2007, which more specifically pertained to online dating. After that, several more papers on recommenders for online dating have been published [5, 6]. Although still being modest in quantity, some research has been done on the speed dating domain which relate to computer intelligence [7, 8, 9]. However, what to date seems to be untouched is how to build recommender systems for speed dating.

2.2 Recommender Systems

Recommender systems are information systems that help users obtain recommendations of artefacts or other people based on the user's preferences. Recent times have seen a rise in research of recommender systems and other machine learning techniques due to a vast increase in data world wide. Due to the massive amount of data available these days, a consequence is the issue of information overload. Therefore, users benefit greatly from services that harvest relevant items or people from the large pool of options. Typical scenarios for recommender systems include suggesting movies based on ratings of past screenings, or assisting users in online shopping by matching user preferences to item attributes.

2.2.1 Cold-start problem

The cold start problem is divided into two branches [10]: the new user cold start problem and the new item cold start problem. The new user cold start problem emerges when the user is entirely new to the system, and has not yet provided any preferences to base the recommendations on. The new item cold start problem emerges in situations where a recently added item has not yet been examined by any users, making it hard for the recommender to predict which users might have a preference for that item. For a speed dating recommender, the issue of cold start is built-in because most people visiting events will be first time visitors.

2.2.2 Ratings

The ratings of users might be either explicit or implicit. Explicit ratings are obtained when users rate items on a defined scale, while implicit ratings are supposed positive responses through observed behavior. In the case of a movie recommendation system, explicit ratings might be the amount of "stars" a user give on a 1-5 scale. Implicit signs of liking might be the act of sharing a certain movie to a friend by using some built in functionality in the interface.

The original dataset used in this thesis includes both explicit and implicit ratings. The explicit ratings are the preferences of participants for certain attributes in partners, such as attractiveness, sincerity, intelligence etc. The implicit ratings are if participants mark "yes" on whether they would like to continue see the other person. The latter does not tell us how much, or in what way, the participant likes the other person – only the fact that they do. In the construction of the recommender system described in this paper, the explicit ratings are used to predict the implicit variable.

2.2.3 Content-Based Filtering

Recommendations coming from a content-based classifier are items that are similar to those that have previously been liked by the user. The classifier learns by finding a set of items in which the user has expressed interest, and builds an item profile for each item. A user profile is then constructed by finding shared characteristics of the item profiles. When the user profile is built, that profile is then matched to unseen items, and those of the unseen items that match the user profile will be used as possible candidates for recommendation. Prediction is made by finding the relationship between the item and the user profile. This is done by calculating their similarity. This can be done by using different techniques, cosine similarity being one of the common. Calculating the cosine similarity is done by placing the user profile vector and the item vector in a two dimensional space, then calculating the angle between the two. A smaller angle between them implies a smaller distance, and therefore a higher similarity. The items with the highest similarity with the user profile are those that will be picked for recommendation.

Because items are recommended based on user profiles built on users' historical ratings, new items does not require having any prior ratings before being put on the market for recommendation. In the context of matchmaking this is obviously desired because people are different, and systems that capture that fact to a large extent will likely be favored by the users. Another advantage of content-based models is their ability to adjust to users with unique tastes. Since recommendations are made based on the interrelation between user and item, more personalised recommendations can be obtained. However, this is at the same time a curse known to content-based approaches because it increases the risk of overfitting. Overfitting occurs when the model gets too specialised on specific instances of interest, and thus too narrow to realize that this might not be the user's entire palette of interests. Another central issue is that the algorithm fails to notice the popularity of items stemming from other people's ratings. Although the system aims to provide highly personalised recommendations, utilizing some degree of crowd sourced judgement is often beneficial. Furthermore, one of the reasons that content-based approaches are not more widely used is that it is hard to identify appropriate features to use. For instance, in the case of music, it is not obvious what features to select.

2.2.4 Collaborative Filtering

Collaborative filtering methods base recommendations on the similarity between user ratings. The similarity could be measured on a user-user basis or between items [11].

In the case of user-user, the system looks at which users have rated certain items similarly. The users that are found to have similar taste are then said to be in the same neighborhood. The size of the neighborhood may vary, defining a level of strictness in variation. Finding neighbors on a user basis is known to become computationally expensive when dealing with larger datasets [12]. Item-based recommendations find similarities between items that the user has previously liked. This is more widely used due to its ability to better capture the notion of having items be the primary source of interest.

	Book 1	Book 2	Book 3	Book 4	Book 5
User 1	2			3	
User 2		5	4		
User 3	1		2		5
User 4		3		1	4

Table 1: A utility matrix showing user ratings for books

Utility matrices are used to represent the ratings of users for different items. Blank spaces imply unknown ratings, and it is the recommender system's duty to figure out these unknown values. In Table 1, users have rated a collection of books on a 1-5 scale.

Collaborative classifiers are inherently domain agnostic, meaning that they work well for domains that are hard to represent by their content [13]. This is because classification is done based on the ratings from users rather than analysing item attributes. Collaborative filtering also naturally carries the possibility of exposing users to serendipitous recommendations due to the collective judgement of items. Paradoxically, this is also a caveat when using collaborative filtering methods because already popular items tend to stay popular when constantly "recycled" across the system, which therefore limits the chance for less popular items to rise up to the surface. In short, collaborative filtering enables the discovery of items not known to a specific user, but limits the discovery of items still unknown to the larger population.

2.2.5 Collaboration via Content

The previous two sections have described the two basic architectures for a recommender system. A third category called hybrid recommenders uses complementary strategies to make use of strengths in algorithms that might be weaknesses in another. For instance, we might have good reasons to use a content-based filtering architecture, but since content-based filtering is prone to overfitting, it would be wise to consider using a hybrid technique that uses a collaborative filtering strategy that mitigates the risk of overfitting. The collaboration via content method by Pazzani [14] is a hybrid technique that uses the content-based user profiles to detect similarities between users. Among challenges that can be mitigated by this technique is the cold start problem.

	Feature 1	Feature 2	Feature 3	Feature 4	Feature y
User 1	2	2	3	1	
User 2	3	5	4	4	
User 3	1	4	2	2	
User 4	1	3	4	3	

Table 2: A utility matrix for collaboration via content

Similar to the utility matrix presented in Section 2.2.4, in collaboration via content a utility matrix is set up to represent users' ratings. In this case, we construct a matrix of users and their features. The matrix is populated with ratings for different features such as the preference for attractiveness or intelligence, and the mission is to predict one of the features, which in our case is the participants' decision for "like" of dating partners.

3 Speed Dating Dataset

The dataset used in this paper was originally published by Fisman et al. [15] where they studied how male and female participants differ in their selection of a partner. In a follow-up study [16], the authors showed how male and female participants differ in their mate selection based on race. These two studies laid a solid ground for working with ethically charged data, to overcome reinforcing structural discrimination in machine learning. Furthermore the dataset contains an unbalanced distribution of positive outcomes, which too makes it an interesting and important machine learning challenge. This particular characteristic is considered in this paper when evaluating classifiers.

The dataset consists of 4189 speed dates collected over a series of 21 events between 2002 and 2004. The participants were students at Columbia University who all had a four-minute date with every participant of the opposite sex. Before attending, all participants filled in a preregistration questionnaire to express their preferences, personal attributes, self conceived features and expectations relating to the dating events. In Section 6.1.2, the fields that are used in this paper are listed. In the end of each date, the attendees were asked to rate their partner on attributes ranging from visual to personal. They would also rate whether they liked the person or not, and if they would like to receive the other person's contact information, which would be handed out if the liking was reciprocal.

3.1 Exploratory Statistics

This section presents some exploratory statistics to get familiar with the specificities of the dataset.

Table 3 shows the amount of speed dates in which participants marked that they wanted their partner's contact details, in the table shown as "likes". The table also shows the total amount of occurences where both parties in the date liked each other, in which case there was a "match". The second part of the table shows the ratio of likes to matches, meaning the proportion of likes that resulted in a match. The table shows that males are more likely to opt for a second date than females. It also shows that 65% of all male "likes" result in a match, whereas female "likes" result in a match in 45% of the cases.

Likes and matches	
Male likes	1989
Female likes	1529
Matches	1380
Ratio likes to matches	
Males	0.65
Females	0.45

Table 3: Likes, matches and ratios of likes to matches

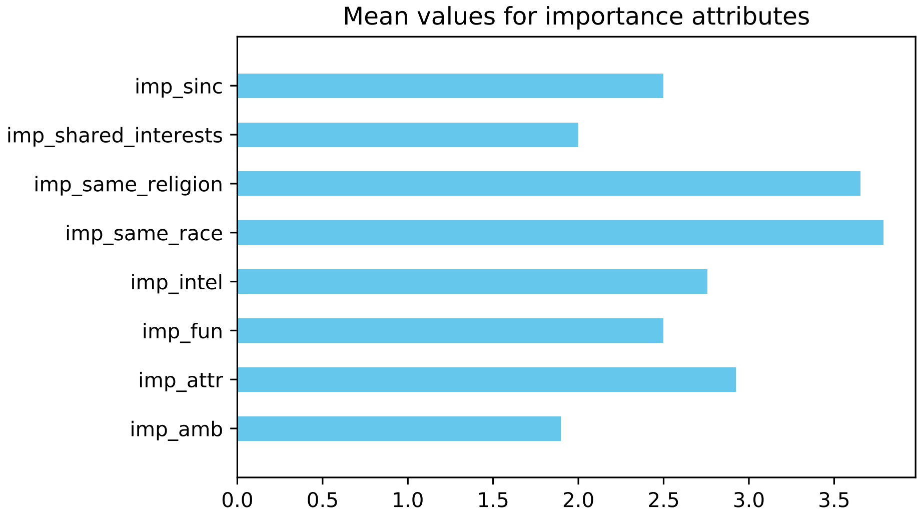


Figure 2: Mean values for importance attributes

Figure 2 shows the mean values for the attributes. In average, race and religion are rated as the most important attributes. However, as is shown in Figure 3 and Figure 4, the most common importance of race for both males and females is the rating 1 of 10. The second highest rated attributes are the importances of partners being attractive and intelligent.

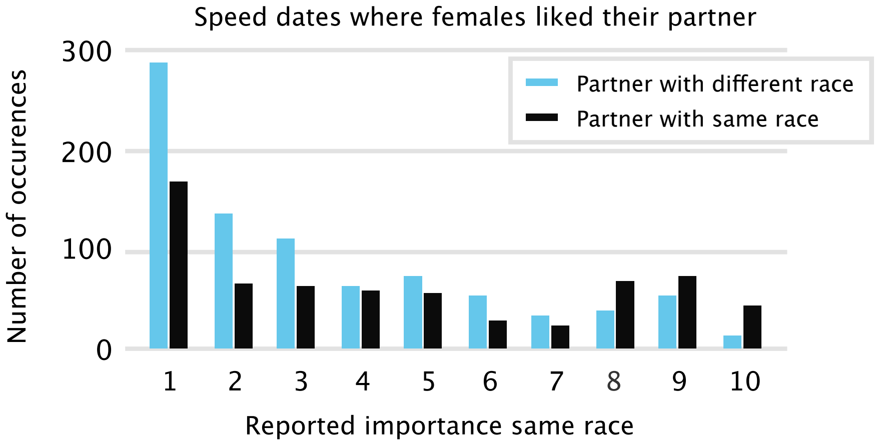


Figure 3: The reported importance of same race from females. The diagram only displays speed dates where females liked their partner

Figure 3 shows a summary of the importance of same race rated by female participants, showing only dates where females liked their partner. The chart shows that the majority of female participants are consistent in their "likes" with regards to their stated preference for importance of dating partners of the same race. However, 22% of the interracial "likes"

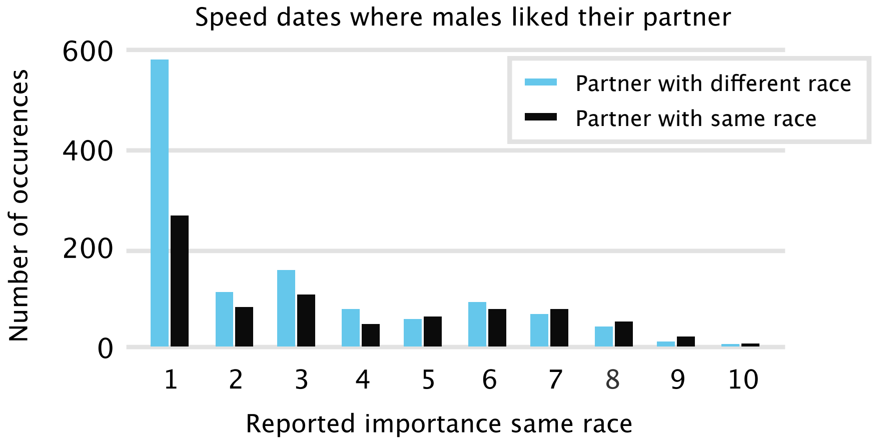


Figure 4: The reported importance of same race from males. The diagram only displays speed dates where males liked their partner

comes from ratings 6 or higher, showing a divergence in stated preference versus outcome in one out of five cases. Similar pattern can be seen in male participants in Figure 4, with the difference being that 18% of the interracial “likes” comes from ratings 6 or higher, showing a slightly smaller divergence in stated preference versus outcome.

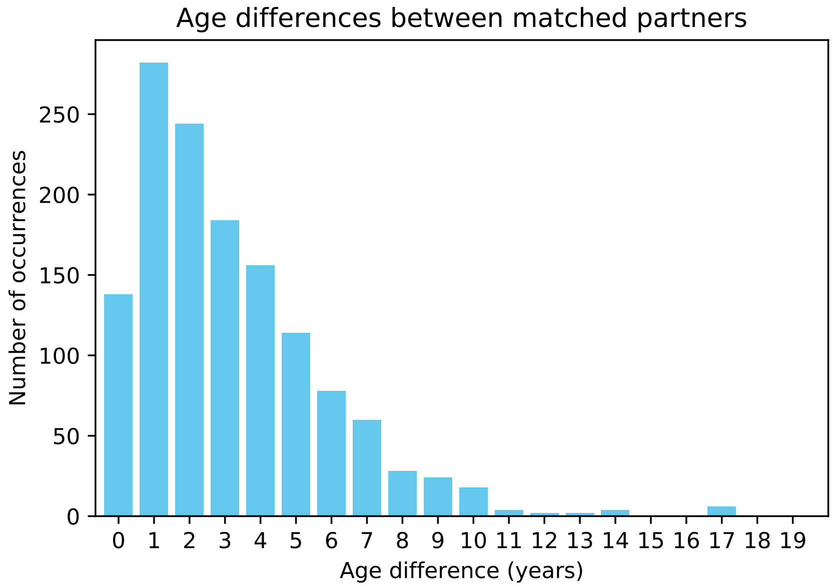


Figure 5: Caption missing

Figure 5 shows the distribution of age differences between matched partners. In 49.4% of the matches, the difference in age is 0-2 years. The maximum difference is 17 years, and the most common is one year, constituting 20.4% of all matches.

4 Related Work

After surveying the literature it became clear that the area of research for speed dating in computer science is still in its inauguration. However, there are some studies fueled by the aforementioned dataset that were found to be useful. Zon [17] undertakes the challenge of building a non-discriminatory classifier, and reports how performance is affected when introducing methods to increase fairness in classification. The paper is a good source of inspiration of how to work with the dataset, which came in to great use when working with this thesis. Ahn et al. [18] explore the importance of feature selection and show how to reason about the dataset from both a theoretically motivated perspective as well as by means of machine learning. The study shows that the machine learning approach outperforms the theoretically backed approach. This indicates that there is an interesting relationship between algorithmic intelligence and the human intuitive knowledge that computers lack.

Using machine learning, Ranganath et. al show in [7] and [8] how to predict social intentions and behavior from spoken conversational features. One of the interesting parts is the ability to predict flirtatiousness and awkwardness. Taking the output from such classifier could be useful in future work for better predicting a match. Still on the topic of predicting behavior in a speed dating context using machine learning, in [19] it is shown how to use features such as position, proximity and motion from video data to predict likings. Combining predictions from classifiers based on these papers could prove useful for creating a database of different features to predict fondness.

Belot & Francesconi [20] did an extensive study on data gathered from 84 speed dating events, hosting a total of 3600 participants. They reported that women "liked" on average 2.6 men, getting a match in 45% of the cases. Men "liked" roughly twice as many women on average, receiving roughly half as many matches [20].

In conclusion, the amount of literature regarding machine learning research for speed dating is very small, and for recommender systems it is nonexistent.

4.1 Similarity With Online Dating

Since the results from speed dating searches were few, a hypothesis was formulated that the searches could be expanded to also include online dating, since they are analogous in some sense. The algorithms in [21, 22] are theoretically motivated because of the logical reasoning behind enforcing reciprocity when predicting matches. However, classification is built upon message passing which does not have any obvious equivalent in speed dating, and the adjustments required were considered too many to be feasible.

[23, 24, 25, 26, 27] uses social network features to build a classifier. Such relational data does not exist for speed dating, and the papers could therefore not be used.

Overall, this partly disproves the hypothesis that the literature found could be used based on the assumption that online dating shares characteristics with speed dating. More precisely it can be said that the literature found at ACM and IEEE could not be used for this study.

5 Methodology

For conducting this report, the design science methodology was found to be a natural way of research. The stated goal with design science is to create a product that is useful for human purposes [28], which is exactly the case here. The created product is called an *instantiation* in the design science parlance [29], which basically means a tangible software system. More specifically the final product here is a proof of concept of a software artefact that shows how to provide speed dating attendees with personalised recommendations of potential matches of their liking.

The process of building a recommender system involves a lot of iterative evaluation to improve the performance, as well as trying different ideas that may or may not qualify for the end result. To make sure that the development process differentiates itself from that of non-scientific software development, Hevner et. al [29] state that both construction and evaluation of the artefact must rely on rigorous methods. It is thus necessary to define specific criterias for success, as well as the means by which those criterias are to be fulfilled. By using the well known F1 score and PR AUC metrics (Section 7), the system is continuously evaluated with the goal of reaching higher scores, meaning more accurate predictions. A minimum level of accuracy is set to beat the case of plain random guessing, or else the system is not of much use.

A central part of the construction of the recommender system is the use of machine learning techniques. Therefore, a survey of methods is made to assess their suitability for the posed problem. The problem is formulated as a binary classification problem with outcome 1 or 0, or *like* or *not like*. Certain classification methods are good for such problem, and they are presented in Section 6.2. After selecting promising techniques they are implemented and tested on the dataset described in Section 3. The result of the implementation of machine learning techniques is a recommender system, which is built using three steps:

1. Preprocessing the data.
2. Building a classifier.
3. Constructing and evaluating recommendations.

These three steps are broken down and use proven techniques derived from either literature or best practices in the area. The first step is called the preprocessing phase, in which the dataset is prepared for building input instances for supervised learning. The second step involves learning a classifier that analyses the dataset and learns how to predict unknown outcomes. Lastly, the task is to provide recommendations to the users of the system and measure the accuracy of the recommendations. The three steps described make up the core procedure that is iterated in the development of the recommender system. To make sure that the process for building the system is reproducible for others, all needed building blocks have dedicated sections in this paper.

For all steps in the development process, the software library scikit-learn[1] is used. Scikit-learn is well known Python library in the machine learning community that implements the most commonly used techniques.

[1] http://scikit-learn.org/

6 Recommender Engine

The construction of a recommender engine may vary in complexity depending on the task at hand. For a simple news recommender it might suffice to recommend the five latest articles, without using machine learning techniques at all. A more advanced recommender might be one that shows the ten most popular products on a retail website. Such solutions are called non-personalised recommenders – which, as the name suggests, are not tailored to serve a particular user. Recommending people to people is naturally highly personalised.

6.1 Feature Engineering

Features are those values that the classifier uses to learn how to predict the output variable. Feature engineering is the art of preparing the raw data into a format that represents the problem domain, and in a way that enables the classifier to interpret the data as well as possible. Feature engineering is one of the most important aspects when building a classifier [30]. The crafting of features greatly determines how well a classifier is able to understand and adapt to correlations in the data.

6.1.1 Data Cleaning

Before constructing the model, a number of steps are needed to clean the data. In most cases, datasets contain some degree of incomplete data, which must be patched by the person working with it. Natural questions that arise when dealing with large datasets is how to handle missing values, deciding whether a variable should be represented as categorical or nominal values, if performing feature scaling is necessary and so on.

In this scenario, a lot of attendees chose not to answer some of the requested fields. A common way of dealing with missing values is to simply replace them with the mean value over all values for that specific feature, which was the chosen method for this study. Not knowing the participants true standpoint, the mean value was selected as an easy way of portraying someone being somewhat fair to everyone who made the active choice of not answering.

In the dataset, some features were using scales 1-10 while others used 1-100. To avoid having some values be more prominent over others, values are re-scaled and centered around 0.

6.1.2 Feature Selection

An initial pass of manual feature selection is done to exclude features that are not relevant to the problem. For instance, it is reasonable to use a participant's preference for attractiveness, while their expectation for the amount of matches for the evening does not relate to the actual prediction. Table 4 contains the manually selected features that were used as the basis for the automated feature selection.

Feature	**Values**	**Description**
gender	Male, Female	Gender
race	Black/African American, European/Caucasian American, Latino/Hispanic American, Asian/Pacific Islander/Asian American, Native American, Other	Race
race_o	Same as above	Race of partner
age	18-55	Age
age_o	18-55	Age of partner
imp_same_race	1–10	Importance same race
imp_same_rel	1–10	Importance same religion
imp_attr	1–100	Importance partner attractive
imp_sinc	1–100	Importance partner sincere
imp_intel	1–100	Importance partner intelligent
imp_fun	1–100	Importance partner fun
imp_amb	1–100	Importance partner ambitious
imp_sha	1–100	Importance shared interests
self_attr	1–10	Perceived self attractive
self_sinc	1–10	Perceived self sincere
self_fun	1–10	Perceived self fun
self_intel	1–10	Perceived self intelligent
self_amb	1–10	Perceived self ambitious
pf_o_attr	1–100	Partner prefer attractive
pf_o_sinc	1–100	Partner prefer sincere
pf_o_intel	1–100	Partner prefer intelligent
pf_o_fun	1–100	Partner prefer fun
pf_o_amb	1–100	Partner prefer ambitious
pf_o_sha	1–100	Partner prefer shared interests
sports	1–10	Interest for playing sports
tvsports	1–10	Interest for sports on tv
exercise	1–10	Interest for exercising
dining	1–10	Interest for dining out
museums	1–10	Interest for museums
art	1–10	Interest for art
hiking	1–10	Interest for hiking
gaming	1–10	Interest for gaming
clubbing	1–10	Interest for clubbing
reading	1–10	Interest for reading
tv	1–10	Interest for watching tv
theater	1–10	Interest for theater
movies	1–10	Interest for movies

Feature	**Values**	**Description**
concerts	1–10	Interest for concerts
music	1–10	Interest for music
shopping	1–10	Interest for shopping
yoga	1–10	Interest for yoga

Table 4: The features used for training

When training the classifier, for each iteration in cross validation (Section 6.3), automatic feature selection is performed to reduce noise in the training data. This can further improve the accuracy of the classifier because using wrong or superfluous features might actually harm the performance. The method used was by fitting a logistic regression model to select the most important features in each fold of the cross validation.

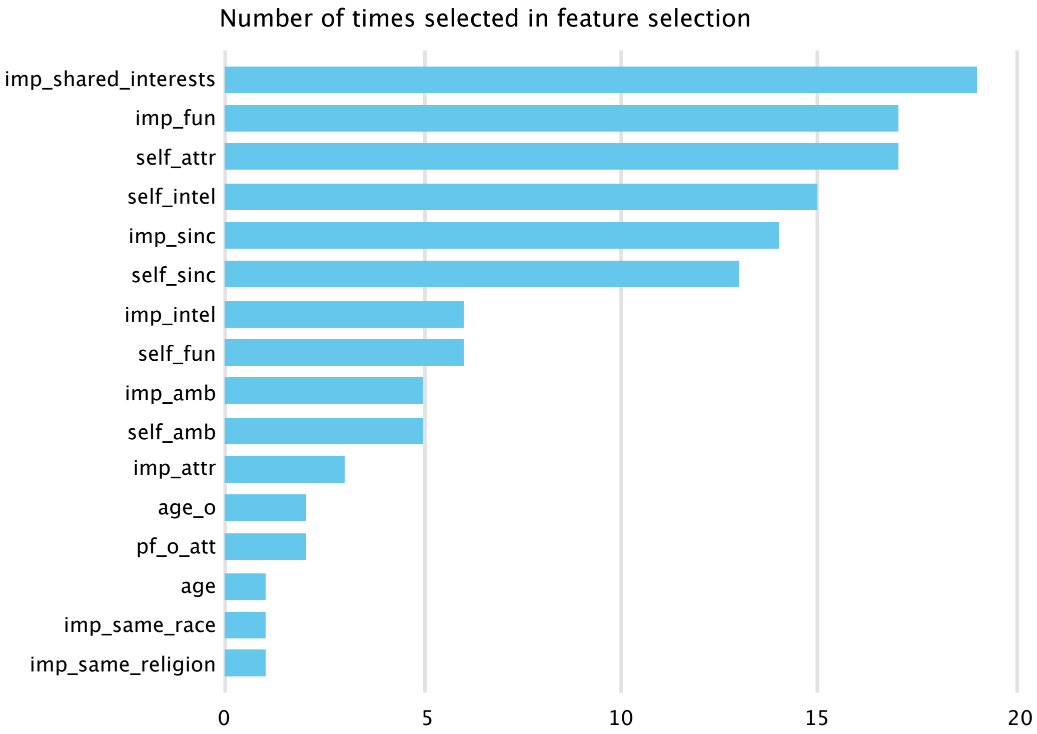


Figure 6: Feature selection

Figure 6 shows a summary of the occurences of features that were selected in the automatic feature selection process. It is shown that the importance for shared interests is most commonly selected feature. In the top is also the preferences for partners to be fun and sincere. The level of one's own perceived attractiveness, intelligence and sincerity also proved to be among the most important features.

6.2 Classification Methods

Classification is the process of predicting what class an item belongs to. In machine learning, classification belongs to the family of supervised learning, since the outcome variable is known beforehand. The classifier learns by being provided a set of training data that is split into instances composed of a feature vector and the associated class that is the target for prediction. The classifier is evaluated by storing a smaller subset of the data as the ground truth, which is then being compared to an equally large set of predictions from the classifier. In addition to the training/test set split, a validation set is allocated from the initial dataset to tune the hyperparameters of the classifier.

Our goal with the classification is to find the probability that someone likes their date, rather than just classifying the outcomes as positive or negative. This is because of two main reasons: we want to rank the candidates, and find a reasonable threshold for the binary classification. The most intuitive threshold is 0.5 because it makes both classes equally probable. However, if one outcome in reality is more likely than the other, we can select a threshold to better reflect that, and thus improve the accuracy of the classifications. The threshold chosen in this case is 0.4199. A more detailed explanation is given in Section 7.2, where it is argued how thresholds affect the F1 score.

6.2.1 Logistic Regression

Logistic regression is a type of linear regression that can be utilized for either classification or regression. Linear regression fits a straight line to the dataset, and outputs a value of y given some value of x. The problem with linear regression is that the fitted line is continuous, and can therefore output values greater than 1 or less than 0. Since our goal is to output probabilities, the output of y must lie in the range 0–1. When the sigmoid function is applied, the model is compressed into an S-shaped line that is confined within the desired range.

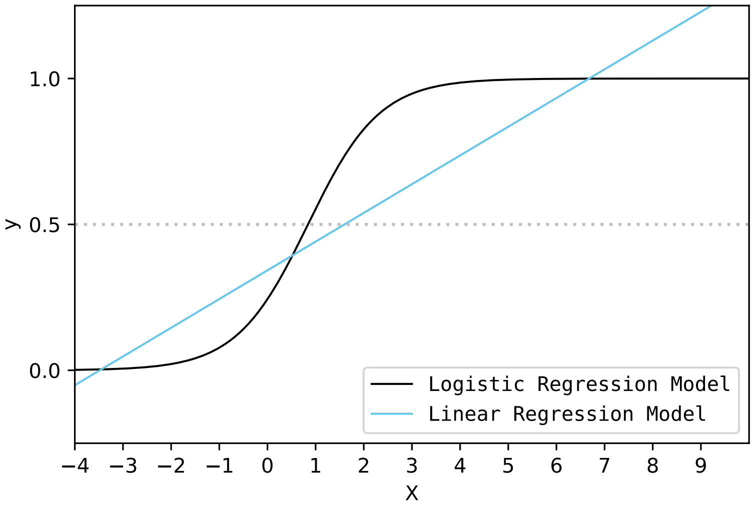


Figure 7: Logistic regression and linear regression model examples

Figure 7 shows an example of a logistic regression model as well as a linear regression model. The dashed line is the baseline that divides the classification into negative and positive decisions. In this example, all predictions above 0.5 are classified as positive, and all predictions below 0.5 are classified as negative.

6.2.2 Random Forest

Random forest is a collection of decision trees, that uses the mean prediction of the individual trees as the output. Decision trees are widely used in data mining, known for their ease of interpretability and prediction speed. However, they are also known for their tendency to overfitting, which is mitigated by using several trees in randomised constellations, known as a random forest.

A decision tree is built by dividing the training data into regions, where each node in the tree splits the data into two subregions. The output value of a prediction is found by traversing the tree and taking the mean value of the observations inside that region. Traversal through the tree is performed by checking the outcome of the condition in each branch until a leaf node is reached, where the decision is stored.

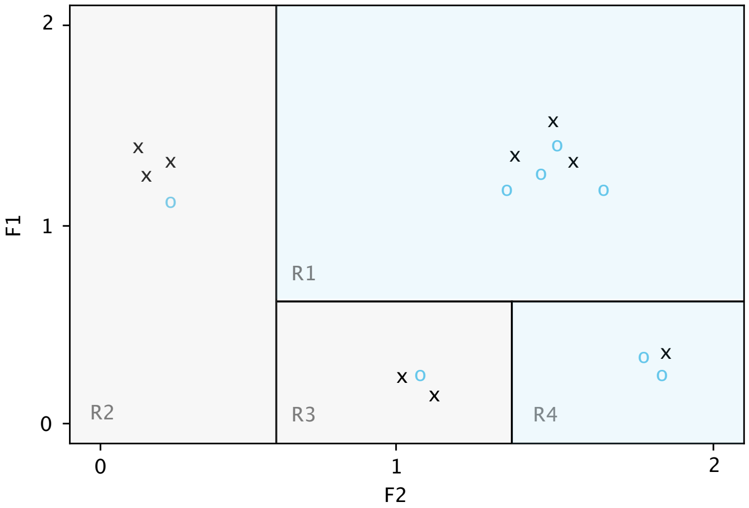


Figure 8: Decision tree example

Figure 8 shows an example of a decision tree classification model. In this example, there are four regions R1, R2, R3, R4. If a new entry e is to be predicted, a traversal might begin with asking whether the condition $F1 > 1$ holds true, in which case we know that e resides in either R1 or R2. Next, if we check the condition $F2 < 0.25$ and get a positive outcome, e belongs in the region R2, and will be classified as a black cross since they constitute a majority over blue circles in that region.

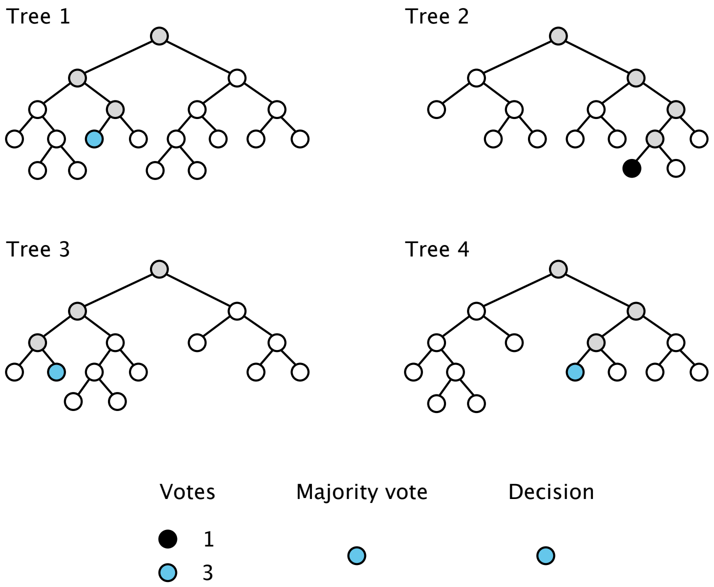


Figure 9: Random forest example

When constructing the random forest, each decision tree is built from a randomly selected sample of the training data. Then, instead of dividing nodes by the best split among all features, as is done with pure decision trees, random forest selects the best split among a random subset of the features.

Figure 9 shows an example of a random forest classification. In the example, four decision trees cooperate to classify an input variable. When a tree in the forest has classified the input variable, it is said to have put a vote on the predicted class. Traversing all trees results in three votes on blue, and one on black. Since the majority of votes was put on blue, that is the predicted class from the random forest.

6.3 Cross Validation

Cross validation means performing classification a number of times to avoid overfitting the model. An arbitrary number of iterations is chosen, and for each iteration the data is split into a training set and a test set. In the speed dating dataset there are 21 dating events, hence giving a natural division of 21 folds for cross validation. For each iteration one event is allocated as the test set, leaving the classifier to learn on the remaining events. When the training of the classifier is complete, the recommender is tested by predicting the “likes” of the real event. To get the final score, the individual scores from the folds are added and averaged.

6.4 Candidate Selection

Candidate selection involves deciding who to choose, and how many of them. It is common to let the classifier rank candidates by the level of probability for success. Then, the amount of candidates can be chosen to be just one if the recommender is serving a mobile application that displays one person at a time. It might be several items if the system recommends a list of books that might interest an active reader. In this case, recalling that the speed dating events accommodated around 20 attendees per night, it is easy to see that the recommender should include all candidate recommendations. This also means that the recommendations are not the top-N candidates as is the most common, because the quantity varies for each user. The candidates chosen with this recommender are the participants that the current participant are predicted to like.

7 Evaluation

Evaluation metrics is a way of measuring the performance of a classifier. Depending on the nature of the problem, which evaluation method is most appropriate varies. In this case the goal is to predict whether an attendee will like their date. It is thus a question of dealing with outcomes where the negative decisions outrun the positive, which is often referred to as an unbalanced or skewed dataset. This section describes the metrics relevant to this paper.

	Predicted negative	*Predicted positive*
Actual negative	TN	FP
Actual positive	FN	TP

Table 5: Confusion matrix

In binary classification there a four possible outcomes metric-wise: two cases for the correct prediction and two for wrong prediction. The correct predictions are called true positives (TP) and true negatives (TN), and the wrong predictions are called false positives (FP) and false negatives (FN). A confusion matrix (Table 5) shows the relation between correct predictions and the faulty ones.

$$Accuracy = \frac{Correct\ predictions}{All\ predictions} = \frac{TP + TN}{TP + TN + FP + FN}$$

The most straightforward way to measure performance is to observe the accuracy of the classifier. Accuracy is calculated by dividing the correct predictions with the total number of predictions. The correct predictions are found by adding the true positives and the true negatives. Using accuracy to predict samples in an unbalanced dataset, however, is not a viable option. Having a skewed distribution means that if 80% of the class labels are negative, we can achieve a high accuracy by simply predicting 80% of the outcomes as negative. For this reason, more sophisticated measurements can be used to better reflect the context.

A common metric is the area under a Receiver Operating Characteristic (ROC) [31] curve. The ROC curve measures how well a model can distinguish between the true positives and negatives. In this case, analysing a model's ability to predict negative samples is not that interesting since the primary focus is to find out who likes who, i.e positive samples. Therefore, a similar metric to ROC AUC can be used, one that calculates the area under a Precision-Recall (PR). The PR curve focuses on the quantity and correctness of the positive predictions.

In addition, the F1 score is used as a single score to measure the balance between the quantity and correctness of the positive predictions.

7.1 Precision-Recall AUC

Precision and recall are two popular metrics to use for evaluating machine learning models. By using them in conjunction we can describe a classifier's ability to pick relevant recommendations.

$$Precision = \frac{Correct\ positive\ predictions}{Positive\ predictions} = \frac{TP}{TP + FP}$$

Precision measures the proportion of correct positive predictions over all positive predictions. Using the precision metric we can measure how good the recommender is at finding positive outcomes over all, and more precisely its ability to find the correct ones. Optimising for precision gives a larger set of correct positive predictions but also more incorrect ones since the recommender is less strict.

$$Recall = \frac{Correct\ positive\ predictions}{Actual\ positive\ predictions} = \frac{TP}{TP + FN}$$

Recall measures the proportion of correct positive predictions over the actual positive predictions. More simply put this is the true positives rate. Finding higher values for the recall measure means that the quality of predictions is higher, but it comes with the cost of finding a smaller amount of correct predictions. In Section 7.2 the F1 score is described, which is a way of balancing the tradeoff between precision and recall.

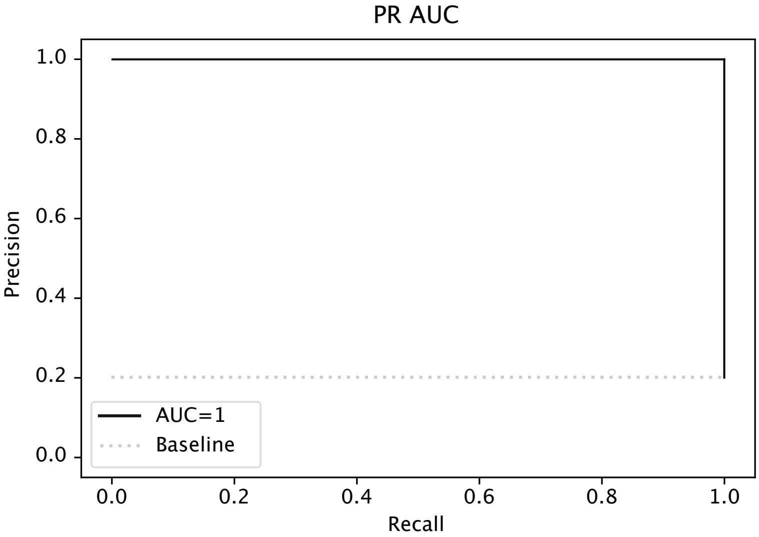


Figure 10: Optimal PR AUC curve with an arbitrary baseline

The AUC is the area under the Precision-Recall curve. The PR AUC plot is a visualisation of the relation between precision and recall over thresholds from 0 to 1. An easy interpretation of the plot is that when the curve tends towards the upper right corner, the better the performance. A perfect curve is a straight line going from the top left corner to the top right down to where the baseline is set. Scores range between 0 and 1, where 1 is the optimal. The baseline of a PR AUC curve is a horizontal line given by $\frac{P}{P+N}$, which is equivalent to the random case.

7.2 F1 score

F1 score is the harmonic mean between precision and recall. As optimising for precision counteracts the recall score and vice versa, it is good to have a single measure of success. Using a score that is consistent with regards both precision and recall also makes it easier to interpret and reason about the results. The F1 score is a value between 0 and 1, the maximum being 1.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

F1 is a metric that works with binary classification, meaning that the probabilities from the classifiers have to be converted into either 0 or 1 using a threshold. A lower threshold typically results in higher precision and lower recall, while a higher threshold yields fewer, more confident predictions. The selection of threshold depends largely on the context, given that certain applications are more fault-tolerant than others. In this thesis, no assumptions are made whether it is more important to be accurate or optimistic. Therefore, the threshold that is used follows the distribution of positives and negatives in the original set, given by $\frac{P}{P+N}$, which is 0.4199.

8 Results

This section reports the empirical results observed from the output of the recommender system. The results show the scores from two trained classifiers as well as their baseline equivalent for both F1 score and PR AUC.

In Table 6 the F1 scores for logistic regression and random forest is presented together with a random classifier that simply guesses 0 or 1 based on nothing.

Figure 11 shows the PR curves for logistic regression and random forest, compared to the static baseline by measuring the area under the curves.

Classifier	**F1 score**
Logistic Regression	0.5038
Random Forest	0.5423
Baseline (Random)	0.4493

Table 6: F1 scores

In Table 6 it is shown that recommendations using the random forest classifier yielded the highest F1 score. The logistic regression model produced slightly worse performance, yet still better than the random case.

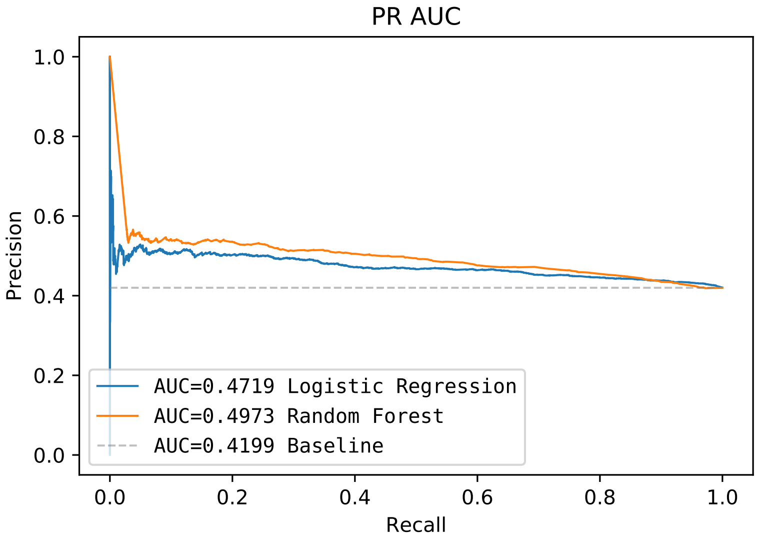


Figure 11: PR AUC plot

The values reported in Figure 11 show that random forest was most performant, with logistic regression being less accurate but better than the baseline.

The results for both metrics show similar tendencies: the random forest model proved most useful, and the logistic regression classifier could be considered theoretically useful.

9 Analysis and Discussion

9.1 Classifier performance

Results show that both logistic regression and random forest do comply to the requirement of predicting the "likes" of attendees at a higher accuracy than by pure chance. It is clear that the random forest classifier is most reliable due to its consistency in performance with regards to both the measurements used.

Although both classifiers perform better than random, the margin by which they do so is fairly narrow. One possible explanation might be found in the crafting of features. As stated in Section 6.1, feature engineering is an essential part of the construction of classification models. It is reasonable to believe that refined feature engineering would lead to more performant models also in this scenario. Instead of building just one model as is done in this thesis, it would likely be beneficial to create several models by deriving features from the existing ones and combine them. This a common practice to create more advanced models, and should be considered when performing future experiments.

The training time of the classifiers was around 20 minutes. In general, speed dating organizers run about a maximum of two events per day. Thus, in a production environment this makes it feasible to use more advanced classifiers that take longer to train, which could be re-trained between events. As is shown in [17], the performance of classification gets reduced when accounting for fairness. If it is possible to trade training time for prediction accuracy, that would be a profitable tradeoff for speed dating recommenders since they do not suffer from any limitations regarding the time that can be allotted to the training of classifiers.

9.2 Feature selection

The features selected in the automated selection process were relatively expected. We can see that participants value a lot of traditionally attractive attributes such as shared interests and that one's partner is attractive, intelligent and sincere.

In addition to this, it was also found that one's own perception of largely the same attributes as the aforementioned ones plays a major role in the likelihood for successful speed dates. The speed dating environment is intensive, and participants meet in a very short time. During these brief meetings, it is important for attendees to showcase their best sides, or at least the ones they consider to represent themselves. Thus, it is not surprising that participants' self-esteem play a significant role.

The results also show that age plays a role in the selection of partners. The average age difference between people in matching dates is 3 years. This indicates that participants in general prefer to date people with similar age as themselves.

9.3 Candidate Selection

Pizzato et al. wrote an influential paper [21] on reciprocal recommenders, where candidate users were selected based on mutual interest as the central driver. Accounting for reciprocity is desirable because it takes both parties' preferences into consideration. In the beginning of this project, the paper from Pizzato et al. was probed to see if it would be viable to base recommendations on reciprocity rather than unilateral interest. Unfortunately, the changes required were found to be too complex to fit within the scope of this project.

9.4 Similarity With Online Dating

Compared with online dating there are similarities as well as major differences. The main similarity is that both dating formats operate on users' definitions of their dream partner, as well as some kind of self-description provided by the users themselves.

When extending the feature set, online dating recommenders tend to use implicit ratings and social network features. Implicit ratings in a digital environment, such as inferring "likes" by counting the messages sent between users, does not seem to have any obvious equivalent in speed dating. However, having access to physical presence enables exploiting other areas of implicit ratings such as detecting flirtatiousness in speech. Such methods could possibly improve the prediction accuracy if integrated into future recommender systems. Worth noting is that collecting implicit data might cause privacy violation concerns, and should be considered when conducting forthcoming experiments.

9.5 Future work

A lot of work is currently being put into machine learning research that focuses on issues regarding fairness and discrimination. As discussed in Section 3, as many as 22% of females and 18% of males that matched with someone of another race, rated 6 or higher on a 6-10 scale regarding the importance of matching with someone of the same race. This implies that about one out of five participants that have strong opinions about dating people of the same race are willing to go on a second date with someone of a different race after meeting them in real life. Unfortunately, it was not possible to account for fairness enhancing techniques within the time frame for this project. Recommending people to people is an ethically charged subject and should seek out best practices to achieve fairness.

10 Conclusions

In this thesis, a method for building a recommender system for speed dating has been presented. Common recommendation methods have been discussed and framed inside the context of speed dating. The existing literature on recommender systems for online dating was found to be insufficient for using as a basis for building a recommender system for speed dating. In this thesis it has been suggested that those are two different areas of research that need separate methods for improving.

Two classification models were trained on a real world speed dating dataset. The classifiers were set to predict the probabilities of speed dating attendees to like other attendees partaking at the same event, in order to present a personalised list of recommendations of promising partners to each visitor. Recommending participants based on unilateral interest was necessary due to the scope of this project, and future experiments should consider developing a method that is based on the existing literature on reciprocity as well as fairness enhancing techniques.

Recommendations were evaluated by measuring F1 score and Precision-Recall AUC. The top performing model for both measurements was the random forest classifier. Although the recommender system performed better than random guessing, the trained models probably need substantial improvements in order to be useful in a real-world setting.

References

[1] A. Smith and M. Anderson, "5 facts about online dating," Feb 2016. Accessed: 2018-08-16.

[2] N. Ambady and R. Rosenthal, "Thin slices of expressive behavior as predictors of interpersonal consequences: A meta-analysis," *Psychological Bulletin*, vol. 111, pp. 256–274, 03 1992.

[3] P. Eastwick and E. Finkel, *Speed-dating: A powerful and flexible paradigm for studying romantic relationship initiation*, pp. 217–234. Taylor & Francis Group, 2008.

[4] L. Brozovsky and V. Petricek, "Recommender system for online dating service," *CoRR*, vol. abs/cs/0703042, 2007.

[5] J. Kunegis, G. Gröner, and T. Gottron, "Online dating recommender systems: The split-complex number approach," in *Proceedings of the 4th ACM RecSys Workshop on Recommender Systems and the Social Web*, RSWeb '12, (New York, NY, USA), pp. 37–44, ACM, 2012.

[6] E. Andrews, "Recommender systems for online dating," *Master Thesis, University Of Helsinki*, May 2015.

[7] R. Ranganath, D. Jurafsky, and D. McFarland, "It's not you, it's me: Detecting flirting and its misperception in speed-dates," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, (Stroudsburg, PA, USA), pp. 334–342, Association for Computational Linguistics, 2009.

[8] R. Ranganath, D. Jurafsky, and D. A. Mcfarland, "Detecting friendly, flirtatious, awkward, and assertive speech in speed-dates," *Comput. Speech Lang.*, vol. 27, pp. 89–115, Jan. 2013.

[9] E. Pierson and A. Suciu, "Predicting speed-date outcomes with conversational and network features," 2011.

[10] L. H. Son, "Dealing with the new user cold-start problem in recommender systems: A comparative review," *Information Systems*, vol. 58, pp. 87 – 104, 2016.

[11] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, (New York, NY, USA), pp. 285–295, ACM, 2001.

[12] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, (New York, NY, USA), pp. 230–237, ACM, 1999.

[13] F. Isinkaye, Y. Folajimi, and B. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egyptian Informatics Journal*, vol. 16, no. 3, pp. 261 – 273, 2015.

[14] M. J. Pazzani, "A framework for collaborative, content-based and demographic filtering," 1998.

[15] R. Fisman, S. Iyengar, E. Kamenica, and I. Simonson, "Gender differences in mate selection: Evidence from a speed dating experiment," *The Quarterly Journal of Economics*, vol. 121, pp. 673–697, 02 2006.

[16] R. Fisman, S. Iyengar, E. Kamenica, and I. Simonson, "Racial preferences in dating," *Review of Economic Studies*, vol. 75, pp. 117–132, 02 2008.

[17] S. van der Zon, "Predictive performance and discrimination in unbalanced classification," *Master Thesis, Eindhoven University of Technology*, Sep 2016.

[18] M. Ahn, A. Badawy, R. Ren, and C. Zeng, "What makes a perfect match? exploring speed dating survey data."

[19] A. Veenstra and H. Hung, "Do they like me? using video cues to predict desires during speed-dates," in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 838–845, Nov 2011.

[20] M. Belot and M. Francesconi, "Can anyone be the one? evidence on mate selection from speed dating," 2006.

[21] L. Pizzato, T. Rej, T. Chung, I. Koprinska, and J. Kay, "Recon: A reciprocal recommender for online dating," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, (New York, NY, USA), pp. 207–214, ACM, 2010.

[22] P. Xia, B. Liu, Y. Sun, and C. X. Chen, "Reciprocal recommendation system for online dating," *CoRR*, vol. abs/1501.06247, 2015.

[23] R. Nayak, L. Chen, and M. Zhang, "A social matching system for an online dating network: A preliminary study," in *2010 IEEE International Conference on Data Mining Workshops(ICDMW)*, vol. 00, pp. 352–357, 12 2010.

[24] L. Chen, R. Nayak, and Y. Xu, "A recommendation method for online dating networks based on social relations and demographic information," in *The 2011 International Conference on Advances in Social Networks Analysis and Mining*, (Kaohsiung, Taiwan), pp. 407–411, IEEE, July 2011.

[25] K. Zhao, X. Wang, M. Yu, and B. Gao, "User recommendations in reciprocal and bipartite social networks–an online dating case study," *IEEE Intelligent Systems*, vol. 29, pp. 27–35, Mar.-Apr. 2014.

[26] C. Dai, F. Qian, W. Jiang, Z. Wang, and Z. Wu, "A personalized recommendation system for netease dating site," *Proc. VLDB Endow.*, vol. 7, pp. 1760–1765, Aug. 2014.

[27] K. Tu, B. Ribeiro, D. Jensen, D. Towsley, B. Liu, H. Jiang, and X. Wang, "Online dating recommendations: Matching markets and learning preferences," in *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, (New York, NY, USA), pp. 787–792, ACM, 2014.

[28] S. T. March and G. F. Smith, “Design and natural science research on information technology,” *Decision Support Systems*, vol. 15, no. 4, pp. 251 – 266, 1995.

[29] A. R. Hevner, S. T. March, J. Park, and S. Ram, “Design science in information systems research,” *MIS Q.*, vol. 28, pp. 75–105, Mar. 2004.

[30] P. Domingos, “A few useful things to know about machine learning,” *Commun. ACM*, vol. 55, pp. 78–87, Oct. 2012.

[31] T. Fawcett, “An introduction to roc analysis,” *Pattern Recogn. Lett.*, vol. 27, pp. 861–874, June 2006.