DEGREE PROJECT IN TECHNOLOGY,
FIRST CYCLE, 15 CREDITS
STOCKHOLM, SWEDEN 2017

Modelling an individual's selection of a partner in a speed-dating experiment using a priori knowledge

ARTEM LOS

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF COMPUTER SCIENCE AND COMMUNICATION

Modelling an individual's selection of a partner in a speed-dating experiment using a priori knowledge

ARTEM LOS

Bachelor in Computer Science
Date: June 2, 2017
Supervisor: Kevin Smith
Examiner: Örjan Ekberg
Swedish title: Modellera en individs val av partner i ett speed-dating experiment med a priori kunskap
School of Computer Science and Communication

Abstract

Speed dating is a relative new concept that allows researchers to study various theories related to mate selection. A problem with current research is that it focuses on finding general trends and relationships between the attributes.

This report explores the use of machine learning techniques to predict whether an individual will want to meet his partner again after the 4-minute meeting based on their attributes that were known before they met. We will examine whether *Random Forest* or *Extremely Randomized Trees* perform better than *Support Vector Machines* for both *limited attributes* (describe appearance only) and *extended attributes* (includes answers to some questions about their preferences).

It is shown that Random Forests perform better than Support Vector Machines and that extended attributes give better result for both classifiers. Furthermore, it is observed that the more information is known about the individuals, the better a classifier performs. Clubbing preferences of the partner stands out as an important attribute, followed by the same preference for the individual.

Sammanfattning

Speed dating är ett relativt nytt koncept som tillåter forskare att studera olika teorier relaterade till val av partner. Ett problem med nuvarande forskning är att den fokuserar på att hitta generella trender och samband mellan attribut.

Den här rapporten utforskar användning av maskinlärningsteknik för att förutsäga om en individ kommer vilja träffa sin partner igen efter ett 4-minuters möte baserat på deras attribut som var tillgängliga innan de träffades. Vi kommer att undersöka om *Random Forest* eller *Extremely Randomized Trees* fungerar bättre än Support Vector Machine för både *begränsade attribut* (beskriver bara utseende) och *utökade attribut* (inkluderar svar på några frågor om deras preferenser).

Det visas att Random Forest fungerar bättre än Support Vector Machines och att utökade attribut ger bättre resultat för båda klassificerarna. Dessutom är det observerat att ju mer information som finns tillgänglig om individerna, desto bättre resultat ger en klassificerare. Partners preferens för att besöka nattklubbar står ut som ett viktigt attribut, följt av individers samma preferens för individen.

Contents

List of Figures

Chapter 1

Introduction

1.1 Purpose

Speed dating is a relatively new concept where participants are encouraged to meet many potential romantic partners for a short period of time and later state if they would like to meet them again [1]. This gives researchers the ability to study and confirm various theories related to mate selection. The problem with current research is that it focuses on finding general correlation between a set of attributes and the decision to prefer a partner. This is problematic because it does not consider the individual and his specific preferences. Thus, we want to investigate if we can develop a model that can predict whether a person will want to meet the partner again after their first 4-minute meeting based on data from one speed-dating experiment. Our idea is to use machine learning techniques and a priori knowledge about the candidates. We want to compare if *Random Forest* or *Extremely Randomized Trees* perform better than *Support Vector Machine* and examine if we can increase performance by using hyperparameter optimisation and sequential feature analysis.

1.2 Research Question

The goal of this report is to be able to examine the extend that machine learning techniques can be used to predict if an individual wants to meet his partner again after a 4-minute meeting by comparing the performance of randomized tree methods (i.e. Random Forest or Extremely Randomized Trees) and SVM, hence the question:

> Do Random Forest or Extremely Randomized Trees perform better than SVM with either linear, polynomial or radial basis function kernels when predicting whether an individual will want to meet his partner again after a 4-minute meeting using only information that is known in advance about each candidate.

1.3 Scope

We will only focus on examining whether Random Forest or Extremely Randomized trees perform better than Support Vector Machines. Our classifier will only use information that is known before the partners have met. The data will be taken from one speed

dating experiment only. We will not attempt to explain how our model works nor seek any general heuristics that constitute an individual's decision making. Although we will explore the importance of features, we will not connect it to psychology or explain it in any way. The performance will be measured with c-statistic (area under the curve in a ROC curve) and our only aim will be to maximize this value.

Chapter 2

Background

2.1 Dataset and Similar works

The dataset has 8378 records that contain information about an individual and his partner as well as whether the individual would like to meet the partner again. Before the speed dating event, everyone need to provide basic information about themselves and rank several activities on a scale 1 to 10. After the 4-minutes meeting, they had to rank their partner and assess if they would like to meet each other again.

This data set is already studied in an existing report [1] from a classical statistical perspective. It can be observed from the submission page [2]. Most of them explore the data gathered after the speed dating event and attempt to draw conclusions of what partners look for. Other studies [3] use the idea of *average ratings* of attributes based on the entire population (similar to collaborative filtering). To sum up, current results are based on knowledge *a posteriori*.

2.2 Speed Dating

The idea behind speed dating is to conduct a series of short meetings with a potential romantic partner [4], typically 4-5 minutes [1][5], in order to determine whether or not they would like to meet again [1]. If we have a *bilateral match*, which occurs if both participants have liked one another, then their contact information is exchanged [1]. This enables researchers to explore various fundamental attraction-related hypothesis [4], such as the *similarity principle* [5] and selection principles used by representatives of each gender [1].

2.3 Attributes Used During Selection

Extensive research has been undertaken in order to determine which attributes affect decision making during partner selection. For example, studies have attempted to explore gender differences in mate selection [1] as well as try to confirm the similarity and reciprocation principles [5].

There does exist a discrepancy in the importance of partner's attributes. The study [1], whose dataset is used in this report, suggests that there is a difference in the way males and females choose their partner. For instance, female look for males that are intel-

ligent whereas males focus more to physical attractiveness [1]. However, newer studies emphasise that both genders may state one set of preferred attributes but make their decisions based on something else, in this case, physical attractiveness (that is, both genders value physical attractiveness equally) [5].

Although there is inconsistency in the literature on the way representatives of each gender choose their partner, it is important to stress that selection of a partner may depend on person's intent, i.e. whether a long or short term relationship is sought, in which case women value physical attractiveness also [1].

2.4 Support Vector Machines

Support vector machine (SVM) is a classifier that is based on the idea of separating a p-dimensional feature space into two halves using a $(p-1)$-dimensional hyperplane [6]. For any unseen point x^*, we will compute $f(x^*)$ given that

$$f(x) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p \tag{2.1}$$

which represents the hyperplane. If $f(x^*) > 0$, the point belongs to class 1 and if $f(x^*) < 0$, x^* belongs to class -1.

Given that the data is separable, there exists an infinite number of hyperplanes. The idea is to pick the hyperplane that is as far away from the test observations as possible (i.e. the margin has to be maximized).

Since data may not be linearly separable, SVMs introduce the idea of a kernel: a way to enlarge the feature space. Kernel is essentially a function that quantifies the similarity of two data points. There are many types kernels, for instance: linear, polynomial and radial.

Apart from changing the kernel, it is possible to use C as a tuning parameter to affect the slack variables. It can be viewed as a cost function that specifies the degree of error (i.e. how many points can be on the wrong side of the margin). The value ϵ_i specifies the degree of error: if $\epsilon_i = 0$, the ith point is on the correct side of the margin, if $\epsilon_i > 0$, it is on the wrong side of margin and if $\epsilon_i > 1$, it is on the wrong side of the hyperplane.

$$\sum_{i=1}^{n} \epsilon_i \leq C \tag{2.2}$$

The parameter C controls the bias-variance trade off: if C is large, more violations are allowed so bias is high but variance is low and a small C leads to low bias but high variance.[6, p. 337-353]

The advantage of using classic SVMs in contrast to methods such as k-Nearest Neighbour is that only the support vectors have to be stored in the model.

2.5 Ensemble Methods

The idea behind ensemble methods is to combine the result (using voting, for example) of many *high-variance* and *low-bias* classifiers and thereby reduce the overall *variance* of the ensemble classifier.

2.5.1 Random Forest

Random Forest is an ensemble method that uses a random subset of features during each split of the tree, which makes the trees more decorrelated. Since each split is not allowed to consider the majority of the features, it ensures that all features get a chance to contribute to the model. [6, p. 319-321]

2.5.2 Extremely Randomized Trees

Extremely randomized trees are similar to random forest but instead they randomize both attribute and cut-point choice. An advantage of the algorithm is computational efficiency. [7]

2.6 Feature Selection

Finding the most important features serves two purposes. First of all, it allows us to determine the factors that affect the decision making when selecting a partner we would like to meet again. However, it also reduces the risk of overfitting, by selecting the features that are truly associated with the data and discarding any noise [6, pp. 242-243].

This can be achieved in two ways: by examining the importance of features using forest of trees [8] or by performing sequential feature analysis, for instance, sequential forward selection algorithm [9]. In first case, we obtain relative importance of all the features whereas in the latter case, the aim is to find the set of features that optimizes a certain metric.

2.7 Hyperparameter Optimization

Many classifiers have parameters that are not directly learnt from the data, but instead have to be provided a priori. [10]. In an SVM, the penalty term C can be varied in order to optimize a certain metric. For specific kernels in an SVM, there are additional parameters such as the degree of the polynomial or the value gamma γ that can be optimized. One way to find the optimal set of parameters is to picture them as an n-dimensional space (n is the number of parameters). We can then perform a randomized search or exhaustive search, in order to find the tuple that is optimal.

Chapter 3

Method

In order to determine which classifier performs better, the correct data has to be extracted and then split it into training and testing sets. Secondly, a metric that will assess performance of a classifier has to be defined. By using a baseline classifier, it is possible to ensure that the new classifier performs better than a classifier with a simple learning rule.

Once the data is processed, the better performing classifier can be found. It is achieved by first comparing the area in a receiver operating characteristic curve (ROC curve) and the area in the precision-recall curve. The default settings will be used for all classifiers except for one parameter, which is set to `class_weight='balanced'` (in order to compensate for the unbalanced amount of samples from each class).

In order to ensure that each classifier performs at its best, feature importance will be examined and sequential feature analysis will be performed. These methods find the more relevant features, which gives interesting insights in humans' decision making and also helps to reduce the adverse effect of noisy features. The goal is to always maximize the area under the curve (AUC) in a ROC curve. In addition, hyperparameter optimisation will be performed to the parameters that are not learnt from the data.

When the most important set of features is found and the optimal parameters are determined, it will be used to make the classifier better (i.e. larger AUC in a ROC curve). The goal is to explore if SVM are better than the tree-based methods (Random Forest and Extremely Random Trees), so hyperparameter optimisation will use the kernel as a parameter that is optimized, which means that the result will only use one kernel. Similarly, only the better performing tree-based method will be used.

3.1 Data Extraction

There are four steps to format the data and divide it into two classes: positive (person a wants to meet b, denoted as $a \rightarrow b$) and negative (person a does not want to meet b, given that persons have met):

1. **Initialize the data** – The data is initialized using the Pandas package with `ISO-8859-1` encoding.

2. **Separate the data into two classes** – In order to split the data, we iterate through every record and store the relevant features in a matrix. By default, it is assumed that every entry has the class -1, i.e. the person has not met anyone. Once we find

a record that indicates that it has met with someone already in the matrix, we add its data into the row of the partner. That is, the final matrix will have rows such at

$$(f_a^1, f_a^2, \ldots, f_a^k, f_b^1, f_b^2 \ldots, f_b^k, m) \tag{3.1}$$

where f_a^1 indicates feature 1 of person a, k is the number of features used per person and $m = \{1, 0, -1\}$ indicates whether it is a one-directional match ($a \rightarrow b$), no match or that person a has not met anyone.

We are examining two sets of attributes per partner:

- **Limited Attributes** – age, gender, race
- **Extended Attributes** – age, gender, race, field_cd, date, go_out, sports, exercise, dining, museums, art, hiking, gaming, clubbing, reading, tv, tvsports, theater, movies, concerts, music, shopping, yoga.

The definition of each attribute is found in *Appendix A*. All of the attributes being used are either binary or nominal.

3. **Bootstrapping** – In order to increase the number of data points, we can duplicate each entry in the matrix, i.e.

$$(f_a^1, f_a^2, \ldots, f_a^k, f_b^1, f_b^2 \ldots, f_b^k, m) \rightarrow (f_b^1, f_b^2 \ldots, f_b^k, f_a^1, f_a^2, \ldots, f_a^k, m) \tag{3.2}$$

During duplication of the entries, we should always ensure that the duplicate record has the correct match value. $a \rightarrow b$ can be true whereas $b \rightarrow a$ may not.

4. **Remove *null* rows** - Since the data set contains entries from multiple speed dating experiments, field values are missing. In this step, we remove such entries from the matrix as to ensure that the matrix contains only numerical values.

3.2 Data Splitting

The formatted data will be split using the hold-out method with a test set T_{test} of size 0.3 of the total number of points. The seed value is 4711, which will not be changed throughout the experiment as to ensure that all classifiers operate on the same data. The data will be split using the `stratify`, which will preserve the ratio of positive and negative samples in both the training and testing sets.

3.3 Metrics

In order to assess the accuracy of a classifier, two measures will be used: *Receiver operating curve* (ROC curve) and *Precision-Recall curve* (PR curve). Both of these metrics require understanding of *precision* and *recall* concepts, which are explained first.

3.3.1 Precision

Precision is the ability of a classifier to correctly label positive samples as positive [11]. It is defined as

$$\frac{T_p}{T_p + F_p} \tag{3.3}$$

where true positive T_p is the number of positive predictions given that the data point is positive and false positive F_p is number of positive predictions given that the point is negative. A *positive prediction* is when the classifier predicts a data point to be positive.

3.3.2 Recall

Recall is the ability of a classifier to find all the positive samples [12]. It is defined as

$$\frac{T_p}{T_p + F_n} \tag{3.4}$$

where false negative F_n is the number of negative predictions given that the data point is negative.

3.3.3 Receiver Operating Characteristic Curve

An ROC curve provides a way to evaluate effectiveness of a classifier as an alternative to classification accuracy. There are two metrics that can be considered: the ROC curve itself and the *Area Under the Curve* (AUC). [13]

An ROC curve is a two dimensional graph which plots the *true positive rate* over the *false positive rate*. A typical graph will have a straight line (in this report, a dashed line) that indicates the result of a random classifier, i.e. 0.5 chance to classify a point as positive. Any curve that is below the dashed line should be regarded as a bad classifier. The

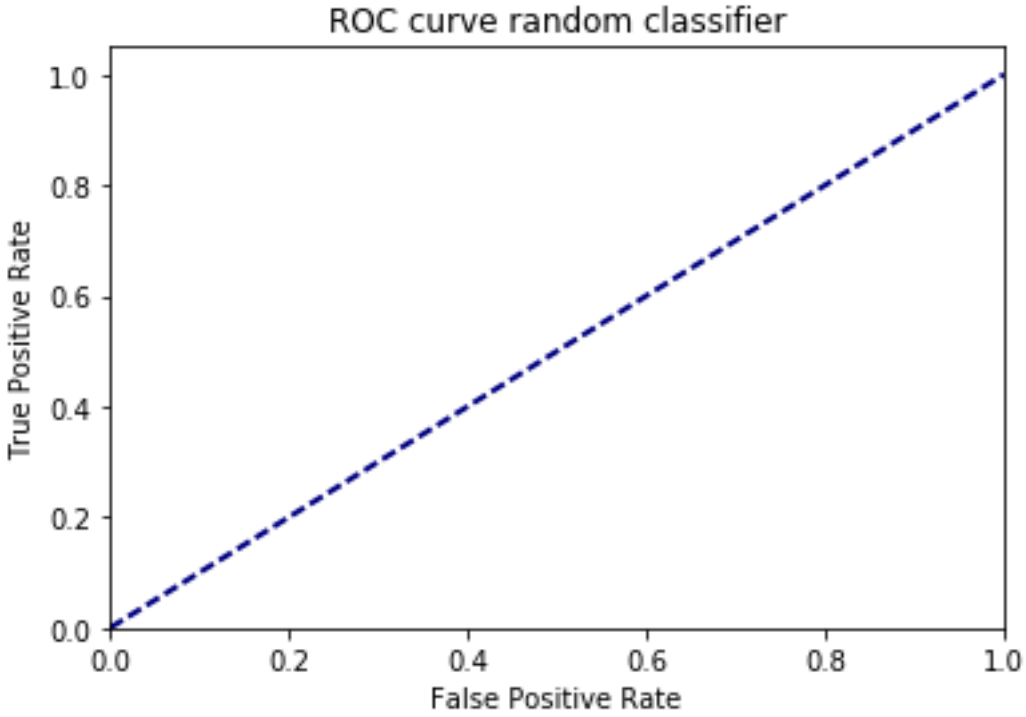

Figure 3.1: An example of a ROC curve with a classifier that classifies a point as positive with 0.5 probability.

aim is to get a curve that is a above the dashed line and that approaches a triangular shape (with the axes as the base and the dashed line as the hypotenuse). For example,

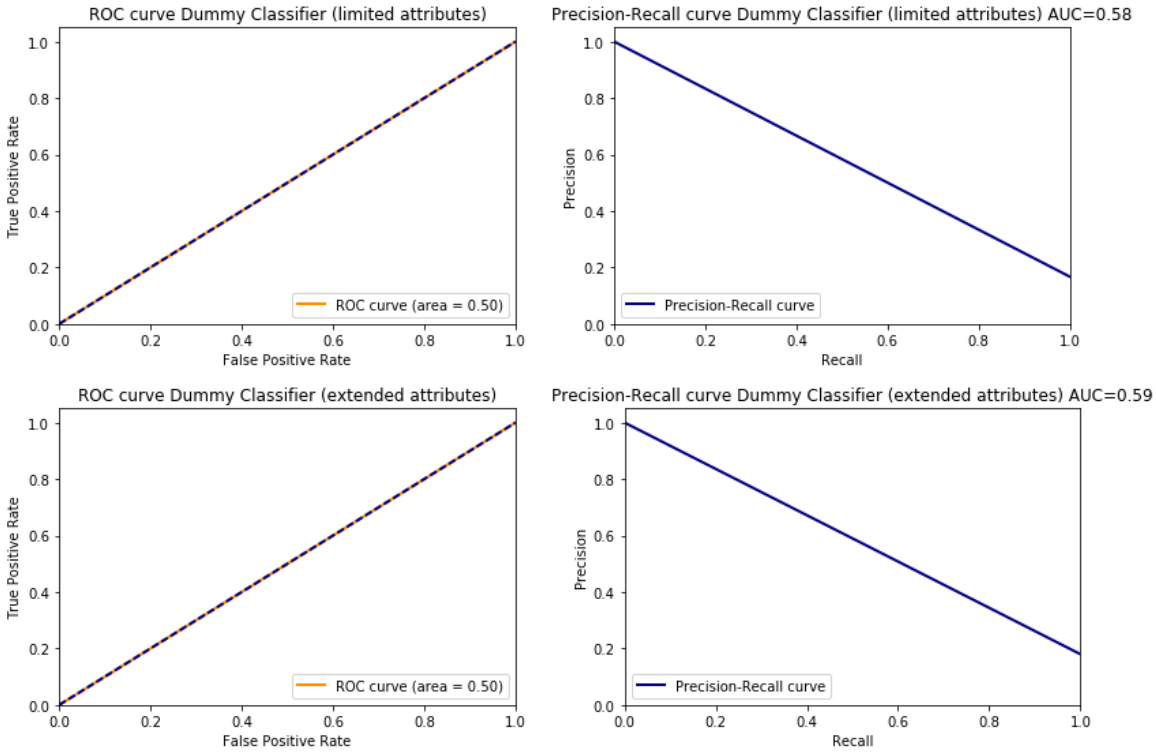

Figure 3.2: ROC and PR curves for the Dummy Classifier with *most-frequent* learning rule.

if we have two classifiers such that one of them is a bit better than random (the curve is above the dashed line) and the other classifier whose curve is above the first classifier, then the classifier with the curve that is above the other superior to the one below.

The second metric is c-statistic (also known as AUC in a ROC curve). The values range from $[0, 1]$, however only $[0.5, 1]$ are of interest since anything below 0.5 is regarded as worse than random. Therefore, the AUC value should be maximized. For example, if we have two classifiers with AUC values $A_1 < A_2$, then A_2 is considered a better classifier than A_1.

3.3.4 Precision-Recall Curve

An alternative measure to ROC curves is *Precision-Recall curve* (PR curve) [14]. They are similar to ROC curves in that that the area can be used to assess the classifier (high area is desired). A high area implies high precision (gives accurate results) and high recall (gives the majority of all the positive results). Since the dataset being used has disproportional amount of positive samples in contrast to the negative samples, PR curves are relevant in our assessment too.

3.4 Base Line Classifier

In order to assess how well a classifier performs, a base line classifier will be used. The strategy is *most-frequent*, which will always predict the most common class (in our case, it is negative). The ROC and PR curves are shown in Figure 3.2.

3.5 Improving Performance of a Classifier

In order to improve performance of a classifier, two approaches will be used: dimensionality reduction to remove noisy features and hyperparameter optimisation to tune the parameters that are not being learnt. In the first case, the sequential forward algorithm will be used. The most relevant features will also be extracted using an extremely randomized tree, although this will not be used directly to improve the classifier. In the latter case, an exhaustive search will be performed over a logarithmic scale for most of the parameters (see Figure 3.5). The choice of logarithmic scale is to make this procedure faster.

Classifier	Tuned Parameters
SVM Linear kernel	$C = \{1, 10, 100, 1000\}$
SVM Polynomial kernel	$C = \{1, 10, 100, 1000\}$, $Degree = \{2, 3\}$
SVM RBF kernel	$C = \{1, 10, 100, 1000\}$, $\gamma = \{1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$
Random Forest	$n\_estimators = \{1, 10, 100, 1000, 10000\}$
Extra Randomized Trees	$n\_estimators = \{1, 10, 100, 1000, 10000\}$

Figure 3.3: Summery of the parameters that are being tuned and their range. All parameters except for the degree use a logarithmic scale.

3.6 Environment

The data collection and analysis are performed in Python 3.6 using *Scientific Python Development Environment* (Spyder). Data extraction in section 3.1 is performed using the *Pandas* package (for step 1) and *Numpy* (in step 2). The experiments are performed using *scikit-learn*. The classification, the data splitting (into test and training sets) and the analysis (ROC curves and Precision-Recall curves) are from this package. For sequential feature selection, `mlxtend` is used.

Chapter 4

Results

4.1 Dataset

Not all data points will be used in the classifier due to null values. The sizes of these sets are illustrated in Figure 4.1.

	Train-set Size	**Test-set Size**	**Total**
Limited Attributes	5732	2457	8189
Extended Attributes	5659	2426	8085

Figure 4.1: The number of data points that will be used when training and validating a classifier.

4.2 Classifier Performance

Performance of each classifier is examined with limited set of attributes and extended set of attributes using the default settings. Only `class_weight='balanced'` is customized. The metrics to assess the classifiers are the c-statistic (area under a ROC curve) and the area under the precision-recall curve.

4.2.1 Limited Attributes

Classifier	**ROC area**	**PR area**
SVM Linear kernel	0.54	0.20
SVM Polynomial kernel (deg=2)	0.56	0.21
SVM Polynomial kernel (deg=3)	0.54	0.18
SVM RBF kernel	0.55	0.18
Random Forest	0.55	0.20
Extra Randomized Trees	0.54	0.19

Figure 4.2: Summary of ROC curve and PR curve areas for different classifier and attribute sets (using limited set of attributes).

4.2.2 Extended Attributes

Classifier	ROC area	PR area
SVM Linear kernel	0.48	0.15
SVM Polynomial kernel (deg=2)	0.65	0.28
SVM Polynomial kernel (deg=3)	0.65	0.27
SVM RBF kernel	0.64	0.26
Random Forest	0.62	0.26
Extra Randomized Trees	0.63	0.29

Figure 4.3: Summary of ROC curve and PR curve areas for different classifier and attribute sets (using extended set of attributes).

4.3 Feature Selection

In order to select the most important features, two methods are used. First, these features are extracted from an extremely randomized tree with `n_estimators=1000` and `random_state=0`, which is illustrated in Figure 4.4 for the limited attributes and Figure 4.7 for the extended attributes. Then, using sequential feature selection, the set of important features that maximize ROC AUC is selected.

4.3.1 Limited Attributes

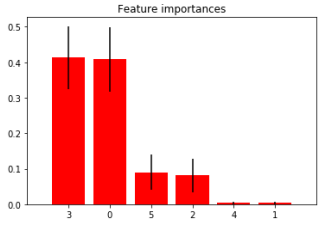

Figure 4.4: Feature importance using an extra randomized tree on the limited set of attributes

Method	Important Features	Max AUC
SVM Linear Kernel	0, 1, 3, 5	0.534
SVM Polynomial Kernel (deg=2)	0, 1, 2, 3, 5	0.570
SVM Polynomial Kernel (deg=3)	0, 1, 2, 3, 5	0.570
SVM RBF Kernel	0, 1, 2, 3, 5	0.625
Random Forest	0, 2, 3, 4, 5	0.917
Extremely Randomized Trees	0, 1, 2, 3, 5	0.920

Figure 4.5: The combination of features that maximizes the area under the curve in a ROC curve on the training set using different classifiers for the limited set of attributes.

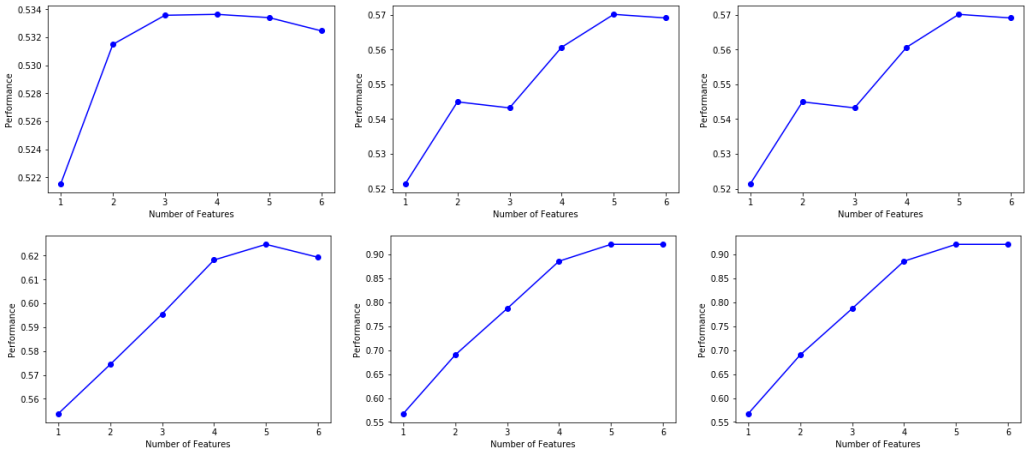

Figure 4.6: The ROC AUC value vs. the number of features for 6 classifiers. From top left corner: SVM Linear Kernel, SVM Polynomial (degree=2) Kernel, SVM Polynomial (degree=3) Kernel, SVM RBF Kernel, Random Forest, Extra Randomized Trees.

4.3.2 Extended Attributes

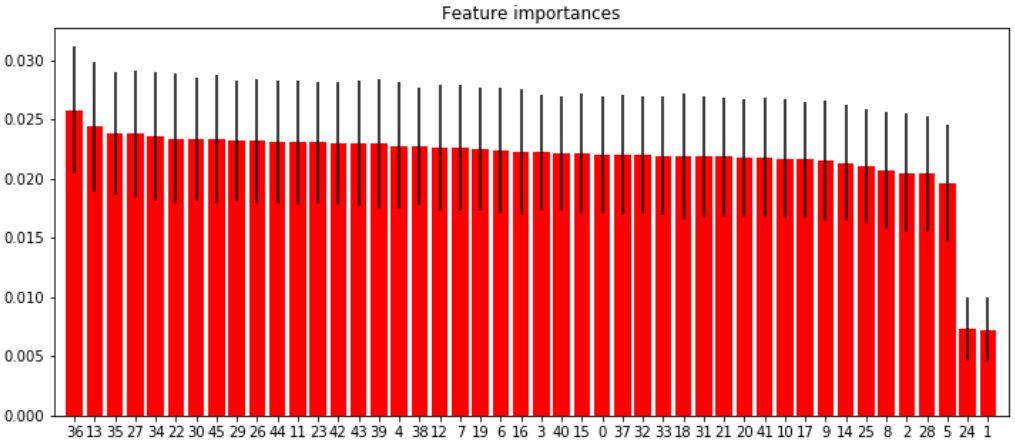

Figure 4.7: Feature importance using an extra randomized tree on the extended set of attributes

Method	**Important Features**	**Max AUC**
SVM Linear Kernel	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 44, 45	0.615
SVM Polynomial Kernel (degree=2)	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45	0.831
SVM Polynomial Kernel (degree=3)	0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45	0.978
SVM RBF Kernel	0, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45	0.956
Random Forest	0, 33, 45, 15, 16, 22, 23, 26	1.000
Extremely Randomized Trees	0, 33, 2, 45, 16, 22, 23, 26	1.000

Figure 4.8: The combination of features that maximizes the area under the curve in a ROC curve on the training set using different classifiers for the extended set of attributes.

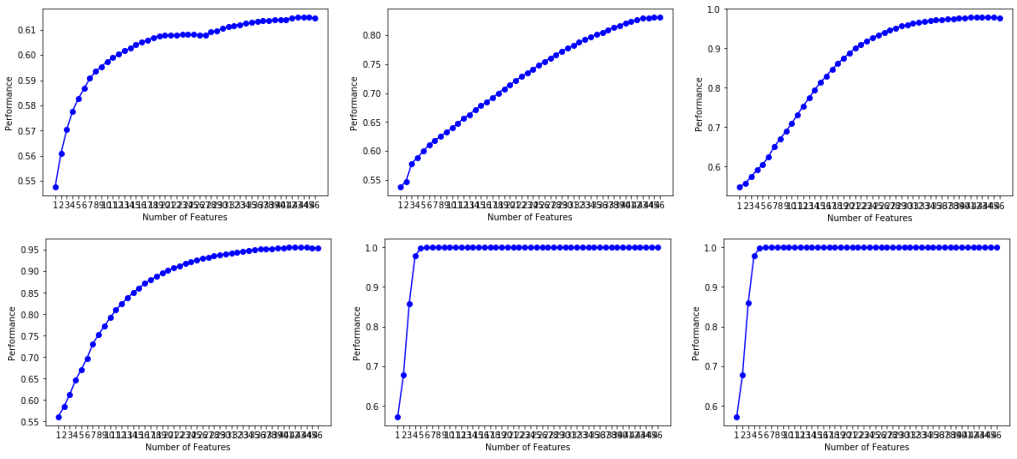

Figure 4.9: The ROC AUC value vs. the number of features for 6 classifiers. From top left corner: SVM Linear Kernel, SVM Polynomial (degree=2) Kernel, SVM Polynomial (degree=3) Kernel, SVM RBF Kernel, Random Forest, Extra Randomized Trees.

4.4 Tuning Parameters

Since the extended attributes perform better for most of the classifiers, the limited attributes will be omitted. Instead, the focus is on extended attributes. The optimal parameters for SVM are shown in Figure 4.10.

Parameter	Optimal Value
kernel	rbf
C	10
γ	10^{-2}

Figure 4.10: Hyperparameter optimisation of SVM

The optimial paramters for the tree-based method are shown in Figure 4.11. Although both random forest and extremely randomized trees had the same AUC value in a ROC curve, the AUC value for random forest was bigger in a precision-recall curve, hence random forest was selected.

Parameter	Optimal Value
Method	Random Forest
$n\_estimators$	1000

Figure 4.11: Hyperparameter optimisation of the tree-based methods.

It was found that dimensionality reduction using sequential forward analysis did not improve the performance in the case of SVM and reduced performance in the case of tree-based methods (see Figure 4.4 and Figure 4.13).

Classifier	ROC area	PR area
SVM RBF kernel	0.66	0.28
Random Forest	0.71	0.35

Figure 4.12: Summary of ROC curve and PR curve areas for different classifier and attribute sets (using extended set of attributes).

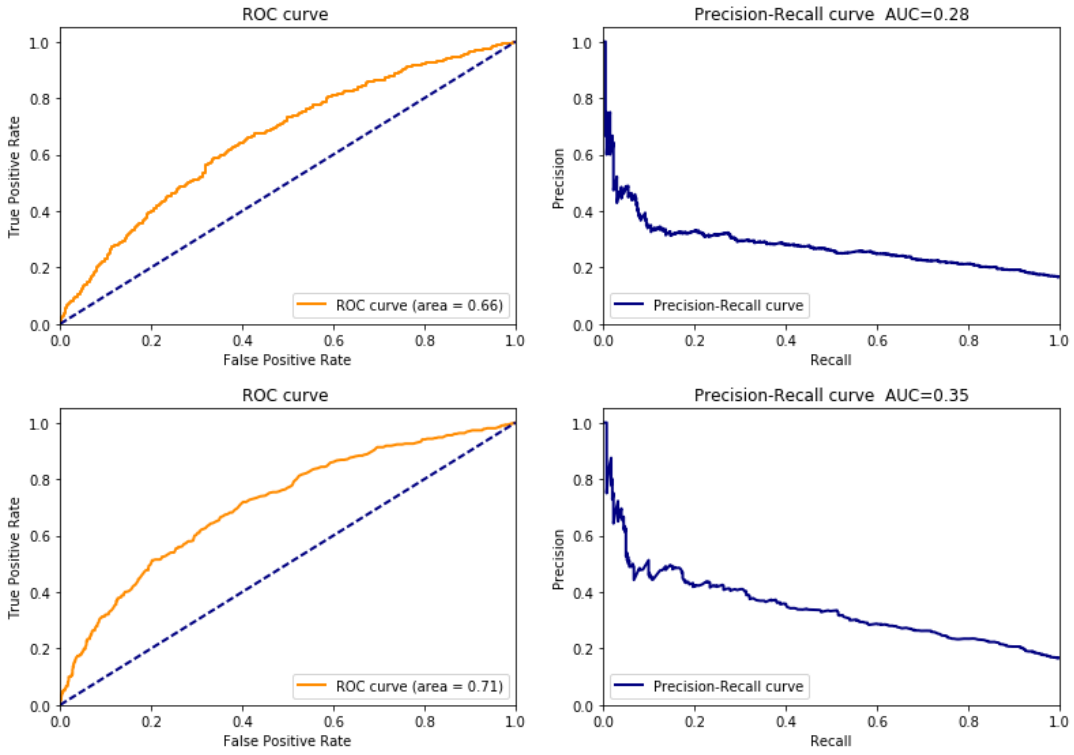

Figure 4.13: The improved classifiers using hyperparamter optimisation and sequential forward analysis. The graphs on top are related to SVM (with RBF kernel) and the ones at the bottom to random forest.

Chapter 5

Discussion and Conclusion

5.1 Best Performing Classifier

The best performing classifier is Random Forest with `n_estimators=1000` operating on extended attributes as shown in Figure 4.13. On the second place, we have the SVM with a radial basis function kernel.

When only six features (limited attributes) are known about each person, classifiers perform worse than having 23 of them (extended attributes). As shown in Figure 4.2, the classifiers are barely above the baseline for limited attributes. However, when more information is known, most of the classifiers perform better (see Figure 4.3). This claim is supported by Figure 4.6 and Figure 4.9, where it can clearly be seen that an increase in the number of features improves the ROC AUC value on the training set.

To sum up, Random Forest performs better than SVM at modelling the decision making when selecting a partner. Generally, the more we know about each person, the better we can predict their actions.

5.2 Important Features

If the only information we have are limited attributes, then age of both partners is an important feature (see Figure 4.4). On the second place, we have the race and finally the gender (which does is not important at all, possibly due to the bias that all people were looking for heterosexual relationships).

If we use the extended attributes, most of the features are equally important (except for gender) as seen in Figure 4.7. Only *clubbing* appears to be more important. If we want to predict if a will like b, then the answer to whether person b likes clubbing plays an important role, followed by a's answer to the same question.

5.3 Limitations

The are several important limitations. First, the dataset is biased since most of the participants are college students. Moreover, it is assumed that students are looking for heterosexual relationships (which is suggested by Figure 4.4 and Figure 4.7, where gender is clearly not important). Thirdly, the experiment is performed in USA, which adds a cultural bias.

5.4 Future Research

In future studies, an option is to explore artificial neural networks (ANN). Although out-of-scope for this report, we got good ROC AUC values (in the range of 0.6) when using Perceptron, without tuning the parameters. By tuning the parameters, ANN can be a faster and better performing classifier for this task. Another option is to keep optimizing Random Forest; this method has more parameters that can be tuned than those that we used, which can potentially increase performance.

Bibliography

[1] F. R. et al., "Gender differences in mate selection: Evidence from a speed date experiment," *The Quarterly Journal of Economics*, 2006.

[2] A. Montoya, "Speed dating experiment | kaggle." https://www.kaggle.com/annavictoria/speed-dating-experiment, Last accessed 2017-03-01.

[3] "Speed dating experiment | kaggle." http://lesswrong.com/lw/lfa/methodology_for_predicting_speed_dating/, Last accessed 2017-03-29.

[4] E. J. FINKEL, P. W. EASTWICK, and J. MATTHEWS, "Speed-dating as an invaluable tool for studying romantic attraction: A methodological primer," *Personal Relationships*, vol. 14, no. 1, pp. 149–166, 2007.

[5] Z. G. Luo S, "What leads to romantic attraction: Similarity, reciprocity, security, or beauty? evidence from a speed-dating study," *Journal of Psychology*, 2009.

[6] J. G. et al, *An Introduction to Statistical Learning*. Springer, 2015.

[7] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp. 3–42, 2006.

[8] "Feature importances with forests of trees." http://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html, Last accessed 2017-05-13.

[9] "Sequential feature selector - mlxtend." http://rasbt.github.io/mlxtend/user_guide/feature_selection/SequentialFeatureSelector/#overview, Last accessed 2017-05-13.

[10] "Tuning the hyper-parameters of an estimator." http://scikit-learn.org/stable/modules/grid_search.html#grid-search-tips, Last accessed 2017-05-13.

[11] "Precision score." http://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html, Last accessed 2017-04-24.

[12] "Recall score." http://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html, Last accessed 2017-04-24.

[13] T. Fawcett, "An introduction to {ROC} analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861 – 874, 2006. {ROC} Analysis in Pattern Recognition.

[14] "Precision-recall." http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html, Last accessed 2017-04-28.

Appendix A

Feature Representation

The list below contains the attribute definitions that are being used for each partner [2]. Figure A.1 contains the feature vector definitions that are used to train a classifier.

- **age** – the age
- **gender** – Female=0; Male=1
- **race** – Black/African American=1; European/Caucasian-American=2; Latino/Hispanic American=3; Asian/Pacific Islander/Asian-American=4; Native American=5; Other=6
- **field_cd** – 1= Law; 2= Math; 3 = Social Science, Psychologist; 4= Medical Science, Pharmaceuticals, and Bio Tech; 5= Engineering; 6= English/Creative Writing/ Journalism; 7= History/Religion/Philosophy; 8= Business/Econ/Finance; 9= Education, Academia; 10= Biological Sciences/Chemistry/Physics; 11= Social Work; 12= Undergrad/undecided; 13=Political Science/International Affairs; 14=Film; 15=Fine Arts/Arts Administration; 16=Languages; 17=Architecture; 18=Other
- **date** – Answer to the question: *In general, how frequently do you go on dates?*: Several times a week=1; Twice a week=2; Once a week=3; Twice a month=4; Once a month=5; Several times a year=6; Almost never=7
- **go_out** – Answer to the question: *How often do you go out (not necessarily on dates)?*: Several times a week=1; Twice a week=2; Once a week=3; Twice a month=4; Once a month=5; Several times a year=6; Almost never=7

The questions below are answers to *How interested are you in the following activities, on a scale of 1-10?*

- **sports** – Playing sports/ athletics
- **exercise** – Body building/exercising
- **dining** – Dining out
- **museums** – Museums/galleries
- **art**– Art

- **hiking** – Hiking/camping
- **gaming** – Gaming
- **clubbing** – Dancing/clubbing
- **reading** – Reading
- **tv** – Watching TV
- **tvsports** – Watching sports
- **theater** – Theater
- **movies** – Movies
- **concerts** – Going to concerts
- **music** – Music
- **shopping** – Shopping
- **yoga** – Yoga/meditation

Feature ID	Short Name	Person
0	age	a
1	gender	a
2	race	a
3	age	b
4	gender	b
5	race	b

Feature ID	Short Name	Person
0	age	a
1	gender	a
2	race	a
3	field_cd	a
4	date	a
5	go_out	a
6	sports	a
7	exercise	a
8	dining	a
9	museums	a
10	art	a
11	hiking	a
12	gaming	a
13	clubbing	a
14	reading	a
15	tv	a
16	tvsports	a
17	theater	a
18	movies	a
19	concerts	a
20	music	a
21	shopping	a
22	yoga	a
23	age	b
24	gender	b
25	race	b
26	field_cd	b
27	date	b
28	go_out	b
29	sports	b
30	exercise	b
31	dining	b
32	museums	b
33	art	b
34	hiking	b
35	gaming	b
36	clubbing	b
37	reading	b
38	tv	b
39	tvsports	b
40	theater	b
41	movies	b
42	concerts	b
43	music	b
44	shopping	b
45	yoga	b

Figure A.1: This table decodes feature id to the short name of the feature for limited set of attributes (left table) and extended set of attributes (right table). The definition of short names is found in *Appendix A*