

The Use of Machine Learning Algorithms for
Adaptive Question Selection in
Questionnaire-based Mental Health Data
Collection Apps

Master's Thesis in Biomedical Engineering
Authors: Alexander Tallroth¹, Max Ålander²
Supervisors: Martin Stridh³, Axel Stålhand⁴

^{1,2,3}Department of Biomedical Engineering, Lund University, Sweden.
⁴ Lytics AB, Malmö, Sweden.

June 21, 2016

Abstract

This report discusses the implementation of machine learning algorithms for personalising question selection in a questionnaire-based self-report app for individuals suffering from mental health issues. A so-called *Random Forest - Genetic Algorithm* (RFGA) hybrid prediction method is used to find an optimal set of relevant questions to pose to new users in the app. A complementary ad-hoc parameter-based question evaluation (PBQE) method which is used to identify questions that are relevant on an individual level, and to control the frequency at which questions are posed to recurrent users is also discussed.

The RFGA method was able to identify a dozen highly predictive questions in a total set of 160 questions, and thus significantly reduce the dimensionality of the feature space. This reduction in the number of features increased the prediction root mean square error of the random forest from 0.11 to 0.15 ($\in [0, 1]$). However, as the main function of the algorithm is to differentiate between features based on their predictive power, and not provide an optimal prediction, this was considered a reasonable trade-off.

The questions that were found to be predictive treated topics of self-esteem, relationships, sleeping habits, attitude towards food, sexuality and physical activity. These questions were formulated in an open way, with simple slider or yes/no answers, and it was found that such questions in general had greater predictive power than more specific questions, for example those with several alternatives.

The PBQE method was able to identify relevant questions on an individual level and increase the probability of these questions being posed to the user, while reducing the frequency at which other, less relevant, questions are posed.

In conclusion, the results show that it is indeed possible to use machine learning methods on mental health self-report data from apps, given that a sufficient volume of high-quality data is available. A key insight is that the formulation and format of the questions greatly affect their predictive power, and that questions should therefore be carefully constructed to be relevant. It was further found that from a machine learning perspective, a smaller number of predictive questions may be desirable in self-report apps, rather than a large number of questions with varying predictive power.

Preface

This project was carried out as a collaboration between Lytics AB, the Department of Social Work at University of Gothenburg, the Department of Biomedical Engineering at Lund University and the social enterprise Karriärskraft in Gothenburg. The authors would like to extend a heartfelt thank you to our mentor Axel Stålhand, who's guidance and support was essential to the success of this project. A thank you also to Mikael Larson and Pål Silow Wiig, whose ambitious and creative vision drives A New Universe forward. We would further like to acknowledge the test group at Karriärskraft for their invaluable feedback, Martin Stridh at the Department of Biomedical Engineering for his support in the writing of this report, Fredrik Jönsson at Lytics for his back-end support and the rest of the team at Lytics for welcoming us and taking good care of us. A final thank you to Expertmakers AB for lending us their offices and lunch room.

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 7 |
| 2 | Background | 11 |
| 2.1 | A New Universe | 11 |
| 2.1.1 | Project background | 11 |
| 2.1.2 | Collecting data | 12 |
| 2.1.3 | The app | 14 |
| 2.1.4 | System infrastructure | 15 |
| 2.1.5 | Users | 16 |
| 2.1.6 | Data composition | 16 |
| 2.2 | An Introduction to some Machine Learning Techniques | 17 |
| 2.2.1 | Decision trees | 17 |
| 2.2.2 | Regression trees | 24 |
| 2.2.3 | From regression trees to random forests | 24 |
| 2.3 | Genetic Algorithms | 25 |
| 2.3.1 | The individual | 28 |
| 2.3.2 | Basic setup of the genetic algorithm | 29 |
| 2.3.3 | Natural selection and evaluation of fitness | 29 |
| 2.3.4 | Selection | 30 |
| 2.3.5 | Mating | 31 |
| 2.3.6 | Mutation | 32 |
| 2.3.7 | Evolution and convergence | 33 |
| 2.4 | The Genetic Algorithm - Random Forest Hybrid | 34 |
| 3 | Method | 37 |
| 3.1 | Goals and Requirements | 37 |
| 3.1.1 | Main goals | 37 |
| 3.1.2 | Improving data quality | 37 |
| 3.1.3 | Meaningfulness | 39 |

| | | |
|----------|---|-----------|
| 3.1.4 | Key requirements | 41 |
| 3.2 | A Brief Summary of the Presented Algorithm | 42 |
| 3.3 | Data Pre-processing | 44 |
| 3.3.1 | In the <i>Random Forest - Genetic Algorithm</i> prediction method | 44 |
| 3.3.2 | In the <i>Parameter-based Question Selection</i> method | 45 |
| 3.4 | Selecting Optimal Questions for New Users | 45 |
| 3.4.1 | Basic setup of the genetic algorithm | 45 |
| 3.4.2 | Fitness | 45 |
| 3.4.3 | Evolution | 47 |
| 3.4.4 | Termination and iteration | 47 |
| 3.5 | Selecting Optimal Questions for Current Users | 47 |
| 3.5.1 | Question parameters | 48 |
| 3.5.2 | Question sets | 49 |
| 3.5.3 | Selecting questions to pose in an individual session | 51 |
| 3.6 | Evaluating the Presented Method | 53 |
| 4 | Results | 55 |
| 4.1 | Selecting Optimal Questions for New Users | 55 |
| 4.1.1 | Dynamics of the genetic algorithm | 55 |
| 4.1.2 | Random forest parameters | 57 |
| 4.1.3 | Selected questions | 57 |
| 4.2 | Selecting Optimal Questions for Current Users | 61 |
| 5 | Discussion | 65 |
| 5.1 | Selecting Optimal Questions for New Users | 65 |
| 5.1.1 | Dynamics of the genetic algorithm | 65 |
| 5.1.2 | Random forest parameters | 66 |
| 5.1.3 | Selected questions | 66 |
| 5.2 | Selecting Questions for Current Users | 69 |
| 5.3 | Ethical Considerations | 71 |
| 6 | Conclusions | 73 |

Chapter 1

Introduction

Data! Data! Data! I can't make
bricks without clay!

*Sir Arthur Conan Doyle (Sherlock
Holmes)*

In a study published in *The Lancet* in 2013, mental unwellness was shown to be one of the main causes of the overall disease burden in the world [1], leading to over 40 million years of disability in 20 to 29-year-olds [2]. Further, depression is one of the main contributors to the burden of suicide and ischemic heart disease [2]. In the UK, only 15 % of individuals that are diagnosed with depression or anxiety receive the full range of treatment recommended by the National Institute of Health and Care Excellence (NICE) [3]. Beyond those who are actually enrolled in a treatment program of some sort, there are many individuals who are not diagnosed and do not seek treatment.

The core characteristic of mental unwellness that makes it a heavy burden on both the individual and on the healthcare system is its close relationship with a wide range of related issues. Poor mental health affects the individual's ability to function in society, leading to alienation and isolation. Some sufferers are not able to work or study, which further affects their condition negatively. Alcohol and drug abuse is common among those affected, and so is violence. Individuals suffering from mental health issues are three times more likely to be abused physically [4]. For women in particular, that number is 10 [4]. These factors, together with the negative affects on physical health mentioned above, make mental unwellness a condition that can affect the sufferer's quality of life negatively for years or even decades. Taking the fact that poor mental health often makes its debut in a very young age into consideration, this becomes a bleak truth.

Mental health issues are heavily under-diagnosed. Many sufferers never seek medical help, or, due to the associated stigmatisation, even tell loved ones about their issues. This makes poor mental health a difficult condition to identify and treat. Often, sufferers receive help when their issues have become so great that significant interventions are required - if they receive help at all.

In Sweden, mental unwellness is very common amongst young women. According to a report by Socialstyrelsen (the National Board of Health and Welfare) in 2009, a third of all young women between 16 and 24 stated that they felt anxiety or depression [5]. This number was 9 % in the 1980's. In this category, norms of society and family situation are key factors that contribute to mental unwellness. Those affected often feel that they have nowhere to turn with their issues, fearing that being honest about their mental health will not be taken seriously.

For young men, norms play an equally prominent role in the extent of mental unwellness. While this group is smaller, with about 14 % of young men in Sweden suffering from mental health issues, the number of unreported cases is believed to be much greater [5]. Sociastyrelsen's study showed that young men often feel that expectations of masculinity are causing them to feel pressured and anxious. These same expectations lead to under-reporting of mental unwellness, effectively creating a downward spiral for the sufferers.

In summary, the negative physical health effects of poor mental health combined with the fact that mental health issues make their debut so early in life and then go unreported makes it such a heavy burden on society and healthcare [6].

The types of treatment available for mental health issues vary greatly globally. In USA, medication is common, with one in ten Americans being prescribed some sort of psychopharmaca, even though the positive effects of such substances vary and the side-effects are many and severe [7]. In Sweden, different types of therapy are more common. Cognitive Behavioural Therapy (CBT) has been shown to have a positive effect on individuals suffering from anxiety or post-traumatic stress, while traditional conversational therapy can help those suffering from light depression [5].

For many of those suffering from less severe mental issues, small interventions have been shown to yield positive results. Socialstyrelsen report that in the 16 to 24 age group, many individuals express their wanting for someone to simply talk to [5]. Primarily, they do not want to be interpreted or analysed, but rather want to speak openly with someone who is supportive, curious and exploring. This is something that has been confirmed in other studies [6] and it is an interesting and important finding, as it suggests that it is a simple showing of interest from another individual that may have the greatest positive effect on those suffering from mild mental health issues. This knowledge has led to novel ways of detecting and treating mental unwellness on a larger scale being explored.

One possible way of providing support for those who suffer from some variation

of mental unwellness is through technology. Today, apps designed for handheld devices have a central role in our everyday lives. We rely on a multitude of these apps to help us find our way to destinations, view videos and photos, communicate with our family and friends, keep track of our physical activities and more. Many of these apps gather a large amount of data on the user - data which can be useful both for the user and the creators of the app. This massive reservoir of data - often referred to as Big Data - can consist of everything from search history to GPS data tracking of our everyday movements. The potential uses of the data are endless. In healthcare, data from patient records and apps is already used in preventive medicine to reduce the pressure on hospitals and primary care units.

A good example of how Big Data can be used to not only increase the quality of healthcare, but also reduce its costs, is presented in a study done by Mount Sinai Medical Centre in Florida, USA on patient readmission prevention. In this study, data scientists created a model based on the data available in the hospital's patient records, for predicting the risk of patient readmission. The model included information on the patients' past admissions, chronic diseases, family history, test results and even socioeconomic factors and produced a list of patients with increased risk of readmission. Doctors at the hospital could then follow up on these patients and keep track of their lifestyle and medication in order to reduce the risk of readmission. In this pilot study, readmissions were more than halved [8]. Not only does this simple change in procedure increase the quality of life for the patients, it significantly reduces costs for the hospital.

In 2013, US healthcare provider Kaiser Permanente hosted a 24 hour data science *code-a-thon* where some interesting findings were made. In this data analysis event which focused on hospitalization due to asthma in USA, data scientists analysed Kaiser Permanente's patient database and found that 17 % of asthma patients do not show up to pharmacies to pick up their medicines and are then 13 % more likely to have an asthma related hospitalization [8]. For a company such as Kaiser Permanente, who provide healthcare to over 9 million people, insights such as this are very valuable, as it allows them to identify issues that lead to further suffering of patients and increased costs. If the number of patients that do not pick up their asthma medicine can be decreased, the number of future hospitalisations can be minimised, and long-term costs reduced significantly.

Given the figures on the costs and suffering related to mental unwellness previously mentioned, gathering and using data on mental health in a similar way to how physiological data is now being used could have an immense impact, both on society and the sufferers themselves. Up until now however, there has been little work done within this field, and it is only in the past few years that scientists have begun to look at data-driven methods for diagnosing and treating mental unwellness. Both novel ways of collecting the data and machine learning techniques for analysing it are being investigated.

A number of commercial apps, such as *Optimism* and *PTSD Coach*, exist today where users can track their mental health through some form of self-reporting. These apps are mainly designed as a support tool for the individual, and data collection is not the main function of the apps. Some feedback and historical tracking is available in some of the apps, but they generally do not make use of machine learning or Big Data analysis to identify more complex patterns in the data. Other apps, such as *Talkspace*, gives the user remote access to therapists and psychologists who can provide support through messaging or by phone, or recommend face-to-face treatment. Such apps focus on the interaction with the therapist and do not gather any additional data.

When it comes to applying machine learning within the field of mental health, focus in recent years has mainly been on identifying biological markers that can be used to explain different types of mental unwellness. These focus of these studies have been to find ways to personalise drug administration to maximise the positive effects while minimising the negative ones. However, in a recent study published in *Molecular Psychiatry* in 2016, scientists explored the possibility of using machine learning techniques on self-report data to predict the severity of major depressive disorder in test subjects [9]. The method developed in this project outperformed conventional regression techniques. These results are interesting, as they indicate that machine learning can indeed be applied to soft mental health data, such as self-reports.

In the project treated in this report, so-called *random forests* and *genetic algorithms* were applied to self-report data from a questionnaire-based app for handheld devices called *A New Universe*. The goal of the project was to investigate whether machine learning techniques such as these can be used to identify questions in the app - and in extension, which topics - that are relevant to identifying the root of the user's mental health issues. This information could be used to pose more relevant questions to the user, and thus improve the meaningfulness of the app and the relevance of the collected data. The resulting algorithm was implemented in a self-report app, and controls the question selection for users in the app.

The report discusses the background to the project, introduces the reader to some machine learning techniques and reflects upon the key requirements of the presented algorithm. Further, it discussed how the requirements are met in the algorithm and presents and discusses results in the form of the dynamics of the algorithm and its ability to differentiate between questions with regards to their relevance. Finally, some ethical considerations of the project are discussed.

Chapter 2

Background

2.1 A New Universe

2.1.1 Project background

A New Universe is an application for handheld devices, developed by Lytics in collaboration with the University of Gothenburg, primarily for the collection of data from individuals suffering from mental health issues. The app has been in development since 2014 and was started as a small project which has grown over the years. The project has been carried out together with Mikeal Larsson, a consultant at Lytics with a background in clinical trials, and Pål Silow Wiig from the Department of Social Work at the University of Gothenburg. The app is currently used exclusively by a small test group at Karriärkraft in Gothenburg. Karriärkraft is a social enterprise that support non-working individuals who wishes to get back into working life and society, and they use A New Universe as part of their work on personal development.

In it's current version, A New Universe interacts with the user by asking a number of questions regarding mental health from a set of 160 such questions. The answers are stored on a server and at the start of this project, the collected data consisted of around 11 000 answered questions.

The long-term goal of A New Universe is to create a new, intelligent system for the collection of large data sets from individuals suffering from mental unwellness. This data may be used for research and allow scientists to draw conclusions on the mental health of larger groups of people and to identify patterns and correlations which today are unknown. Such discoveries may in the future be used in the treatment of mental unwellness. While progress have been made in Big Data analysis of physiological data, equivalent sets of mental health data are not readily available today, nor are efficient methods of collecting it. In this context, A New

Universe may have a significant impact.

Beyond the collection of general mental health data, another possible use of the A New Universe app may be as support in clinical trials or treatments, where it could be used to gain insight into an individual's mental health as he or she progresses through some treatment. In such a scenario, the app could be used to validate the effectiveness of the treatment, while intruding minimally on the patient's everyday life.

With feedback soon being incorporated into the app, the users will be able to track their own progress. Thus, A New Universe may not only be a tool for doctors or scientists to collect data, but could also provide direct support to patients and possibly help them understand and tackle their mental issues more effectively.

The core philosophy behind A New Universe is to move away from conventional clinical trial setups, where carefully constructed and potentially leading questions often are used, and instead pose open questions. The vision is to let the data itself speak, without too much weight being put on individual answers. Lytics, Larson and Silow Wiig believes that continuous data collection using an open question formulation is the next step in how clinical trials are designed and this vision has led Larsson, Silow Wiig and Lytics to consider Big Data and machine learning as ways of interpreting the data collected by A New Universe.

A key concern of the A New Universe project is ensuring the participation of the users. For the app to be successful in collecting large amounts of data, users must be encouraged to regularly interact with the app and to keep using it over longer periods of time. In the case of a clinical trial or treatment this may be easy to achieve, as the patient would have a personal interest in his or her own treatment and progress, and would thus be more inclined to use the app regularly. In the case of large-scale data collection, where the short-term benefits of using the app may not be immediately clear for the user, the problem is non-trivial. Users may consist of individuals with light or passing mental unwellness, who has no contact with a therapist or psychologist, and whose interest in the app greatly depends on a meaningful interaction. If the app is not sufficiently intelligent, the user may quickly lose interest and cease interaction. On the other hand, if the users feel that the app interacts in a way that is relevant for their specific situation, they may make the app a part of their daily life, ensuring long-term data collection.

2.1.2 Collecting data

The core functionality of A New Universe is data acquisition through the posing of questions related to mental health. There are 160 questions in total and each question has a unique ID, is of a specific type and belongs to a certain category. There are 4 types of questions and 13 categories. In table 2.1, the four types are described and in table 2.2, the different categories, their respective IDs and the

number of questions in each category can be viewed. Note that Type 2 questions are no longer used in the app, and are thus not included in table 2.1.

| Type | Description |
|------|--|
| 0 | Yes/No |
| 1 | Slider |
| 3 | Alternatives (multiple alternatives can be selected) |
| 4 | Alternatives (one alternative can be selected) |

Table 2.1: Description of the four different types of questions in A New Universe.

| Category ID | Category name | Number of questions in category |
|--------------|-------------------------|---------------------------------|
| 1 | Alcohol | 14 |
| 2 | General | 28 |
| 3 | Anorexia | 12 |
| 4 | Computer Game Addiction | 5 |
| 5 | Drugs | 10 |
| 6 | Physical health | 15 |
| 7 | Eating habits | 8 |
| 8 | Relationships | 20 |
| 9 | Self esteem | 28 |
| 10 | Violence | 6 |
| 11 | Sexuality | 4 |
| 12 | Gender | 7 |
| 13 | The app | 3 |
| Total | | 160 |

Table 2.2: The different question categories in A New Universe, with the number of questions in each category.

In addition to a question ID, a type and a category, each question has attributes that determine whether it should be asked only once (as is the case with demographic questions), if it should be posed only to users in a certain age group and if it is a mandatory question (which is asked in every session). There are six mandatory questions: "How are you feeling right now?", "How have you been feeling today, on average?", "How many hours did you sleep last night?", "How satisfied are you about what you ate today?", "How satisfied are you about your exercise today?" and "How rested do you feel today?". Lastly, a question can be conditioned on another question, meaning whether it is posed or not depends on the answer to another question.

In the database, type 0 question answers are stored as a zero or a one, depending on the answer; type 1 question answers are stored as a normalized decimal number between zero and one; type 3 question answers are stored as a binary sequence of zeros and ones determined by the alternatives that were selected and type 4 question answers are stored as an integer between one and the number of total alternatives, again determined by the alternative that was selected.

In figure 2.1, examples of questions of four different types are shown. Note that the app is currently only available in Swedish and that the text in the figure is thus in Swedish.

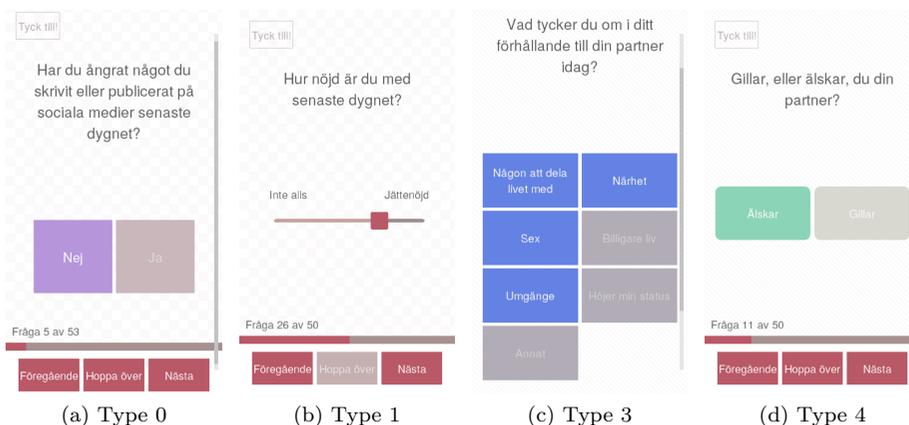


Figure 2.1: Examples of questions of each of the four types in A New Universe. The answers selected in the screenshots would be produce the following data: 0, 0.75, 1110100, 1

2.1.3 The app

In its current version, a set of questions is generated for the user when he or she logs into the app and actively chooses to begin a session. The simple interface of A New Universe allows the user to go back and forth through the questions, and skip questions he or she does not wish to answer.

How questions were generated before the start of this project was a very simple process. Questions are selected at random, with the constraint that questions that have previously been skipped by the user have a lower probability of being selected. The total number of questions selected for a session varies randomly between 15 and 30. This random procedure of selecting questions has had great impact on the characteristics of the data generated so far by A New Universe. This will be discussed in greater detail below.

2.1.4 System infrastructure

A New Universe communicates with a back-end server that handles user information, the questions and the data in the form of question answers. When a user begins a session on his or her instance of A New Universe, the user's ID is sent to the back-end server and a Python script on the server is accessed. This script generates a set of questions, taking the user's simple profile (which contains weights for the questions based on if the user has skipped questions) and the question conditions previously mentioned into account. The set of questions is then sent back to the user's instance of the application. The user's answers to the questions in the set are recorded and sent back to the server to be stored in a database. Figure 2.2 shows the communication flow graphically. It is the question selection component that is the focus of this project.

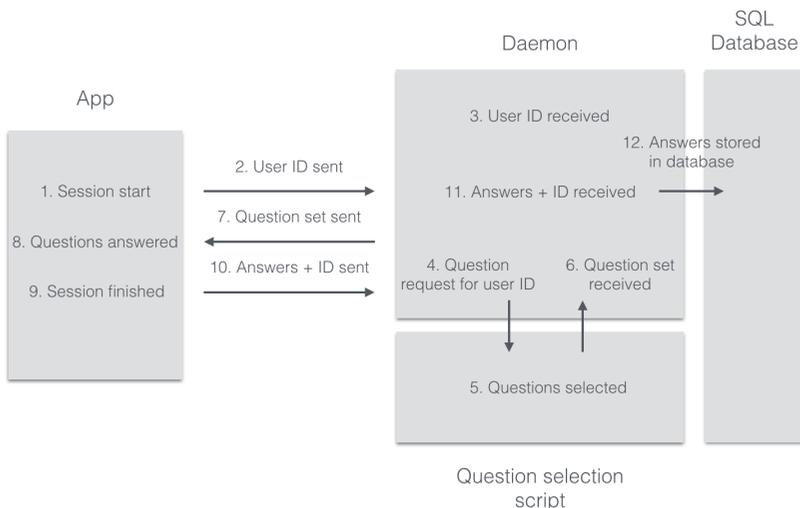


Figure 2.2: The communication flow of A New Universe.

The system uses a PostgreSQL database to store the data and handle the relations between the different variables. In the database, each answer is primarily attributed a user ID and a question ID. The question ID is in extension connected to the question attributes previously described. The answer is also attributed a session, within the user. Thus, from a specific answer, information about which question was posed to which user in which session can easily and quickly be retrieved, along with information on the type and category of the question. Figure 2.3 shows the relationship between variables in the database graphically.

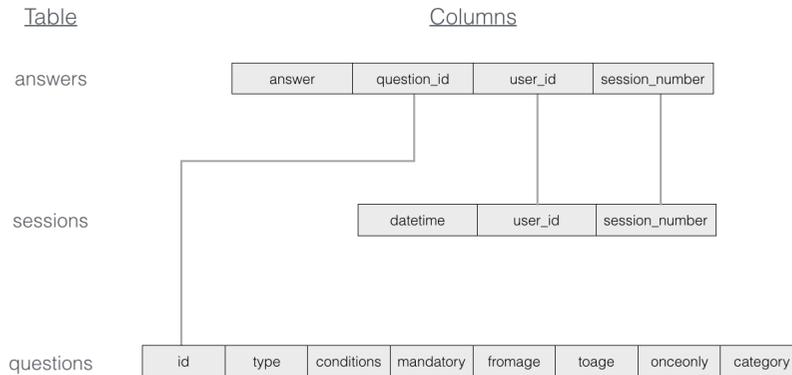


Figure 2.3: The relationships between variables and tables in the SQL database.

2.1.5 Users

As previously mentioned, up until the beginning of this project A New Universe was tested on one primary group: Karriärskraft in Gothenburg. In addition to these test subjects, the app has been used by a small number of employees at Lytics. Participation within these groups varies greatly, with a large number of dropouts. Out of around 80 all-time users, only 20 are currently active (having used the app in the last few weeks). As the main goal with testing so far has been to evaluate the technical and interface aspects of the app, emphasis has not been put on improving participation. Thus, the data that was available at the start of this project was small in size.

2.1.6 Data composition

Out of 160 possible questions, only a small subset of questions are posed to a user in a given session. Thus, for that specific session, data will only be available for a small percentage of the questions. Regardless of how many users are active and how frequently they use the app, for a given user and session the data will be very sparse. Indeed, if one views the data gathered up until the start of this project (which consists of 11007 answers) as a matrix of size $N_{questions} \times N_{sessions}$, where $N_{questions}$ refers to the total number of questions and $N_{sessions}$ refers to the total number of sessions across all users, that matrix is only around 8 % dense, which means that 92 % of all indices are *NaN* or *null*. This characteristic is the core issue that makes conventional data analysis - or even Big Data analysis - difficult in this particular case.

In addition to the general sparsity, data density differs greatly throughout the set: some questions have been posed very few times with many sessions in between, while others are more frequent and clustered in time. Of all questions, only the six mandatory questions have been answered in most sessions. Changes to questions and the addition of new questions over the course of development have further created inconsistencies in the data. Some type 3 questions have had the number of alternatives or the order of the alternatives changed, something which of course corrupts the data. One question has also had its type changed.

Inherent to methods, statistical or other, that examine the relationship between variables is that they are dependant on samples being available for the variables in each data instance. If values are missing, they often have to be imputed from surrounding data. In the case with the A New Universe database, the data is very sparse and would have to be imputed from a small number of samples. If an answer is missing in a session, one might be forced to impute it from a single answer from two weeks earlier, which is, of course, less than optimal. Creating an algorithm that can find and make use of patterns in this sparse data - which is the goal of this project - is thus a challenging task.

2.2 An Introduction to some Machine Learning Techniques

With some background to the project now presented, we proceed to introduce the reader to a few machine learning methods that are used in the presented algorithm. How these methods are used, and how they interact with each other, is discussed in detail in the *Method* section of this report. This introduction is simply intended to familiarize the reader with these tools, so that the method will be easier to follow.

2.2.1 Decision trees

On page 41, a detailed list of requirements and goals of the presented algorithm can be viewed. Without jumping ahead of ourselves, it can be mentioned that “Measuring the relevance of the different questions with some method” is listed as one of the main requirements of the algorithm. In A New Universe, *relevance* of a specific question can be defined as the relationship between the answers to the question and the user’s well-being. For example, if sleep quality heavily affects a person’s life, then one would expect there to be some relationship between answers to the question “For how many hours did you sleep tonight?” and the person’s self-reported well-being.

To someone with a knowledge of statistical methods, the first thing that comes to mind when thinking of possible ways to measure this relationship might be

correlation. Indeed, a modified version of the standard Pearson product-moment correlation [10] is used in one component of the presented method and will be described in greater detail further ahead. First however, a non-linear machine learning method which is also used in the presented algorithm to find subsets of relevant questions is introduced.

In Big Data scenarios, where the relationship between variables is unknown and the scope of the data is large, many classic statistical methods are not optimal - mainly due to their core characteristic of only measuring the linear relationship between two variables. While Linear Regression models [11] can handle multivariate cases, they too are unable to describe non-linear relationships in the regressors. In the case of A New Universe, there are very likely non-linear interrelationships between the questions, something which linear models would not be able to pick up. A method which is becoming increasingly common to use in these types of prediction scenarios is the non-linear *decision tree* predictive model (regression trees, in the case where the target variable is continuous). The concept behind decision trees will now be explained using the well-known iris data set [12].

Fischer's iris data set consists of 150 measurements of four features in three different species of iris flower, collected on the Gaspé Peninsula, Canada, in the 1930's by biologist and statistician Ronald Fischer [12]. The features are the width and length of the sepals and petals, in centimetres. Which of the three species of iris flower each measurement corresponds to was known by Fischer, and this information is included in the original data set as the *target variable*. In this example, one would like to grow a decision tree that predicts the species of iris based on the dimensions of the petals and sepals. This is done by training the tree on a set of features for which the corresponding species is already known, i.e, where the so-called *ground truth* [13] is available. When the tree has been trained, it can then be used to predict the species of iris for a completely new set of feature measurements (where one does *not* know the species). This is a typical machine learning [13] problem.

Firstly, let's study the composition of the data. In figure 2.4, the four features are plotted from a number of different views.

From these plots it is not difficult to see that the features seem to have some structure to them. Depending on the view, one can even surmise that the features seem to be clumped together in groups. It immediately becomes clear from this grouping that there is no strictly linear relationship between the variables.

In figure 2.5, the data is shown from a petal dimensions perspective. In this figure, the grouping is clearly visible. Based on what one can see in this plot, it may be tempting to perform a simple classification by dividing the data into two groups: if the petal length is less than 2.5 cm or the petal width is less than 1 cm, the data point belongs in group 1 and if the petal length is greater than 2.5 cm or the petal width is greater than 1 cm, it belongs in group 2. Indeed, this

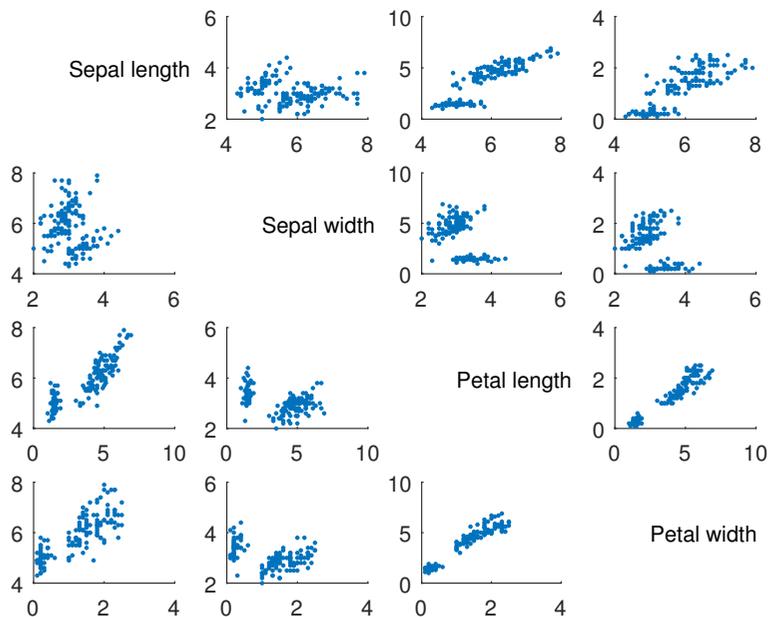


Figure 2.4: The different features of the iris dataset, plotted from various views.

seems to be a reasonable classification, based on the information available on the petal dimensions. However, if one instead studies the data by plotting sepal width against petal width, as is shown in figure 2.6, one will realise that there may very well be more than two groups.

It can be seen that there still appears to be one fairly well-defined group, visible in the left part of the plot, but the composition of the data in the right part of the plot appears to be less regular. In this part of the plot, one can surmise an area of high density, closer to the centre, and to the right of this, an area of lower density. From this perspective, it is less clear where the data set should be split. Ideally, one would like to first split the data based on the two clearly visible groups, and then attempt to split the right group into two additional groups. One would also like to make splits based on different variables, as different views of the data seems to provide different information. This is precisely how decision trees function.

The basic concept of the decision tree method is visualised in figure 2.7. In this figure, the three main components of the tree structure can be viewed: the *root*, the *branches* and the *leaves* [14]. The root and the branches are made up of

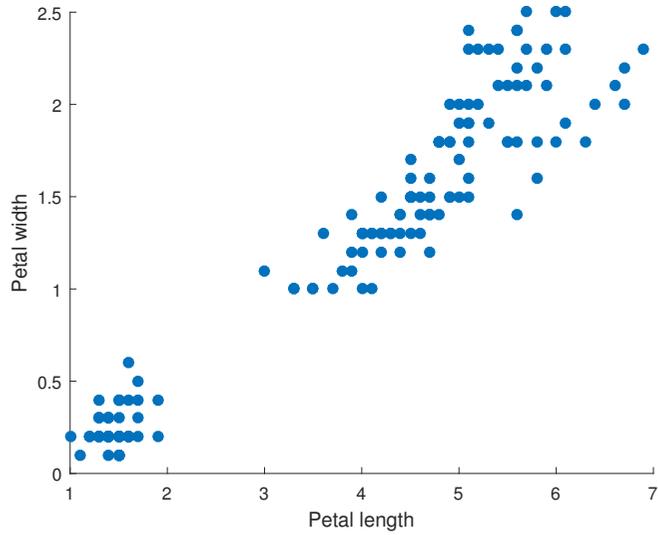


Figure 2.5: Scatter plot of the petal dimensions from the iris dataset.

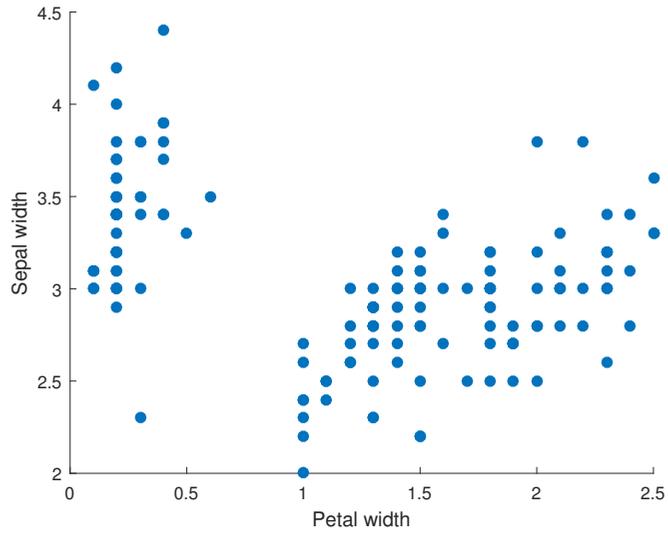


Figure 2.6: Scatter plot of sepal width against petal width.

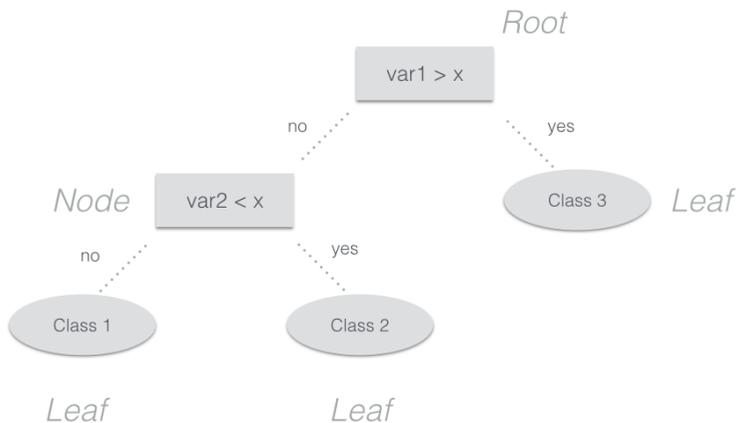


Figure 2.7: The basic concept of a decision tree.

nodes, each of which represents a split of the target variable based on the input variables, or features. Each node can be seen as a boolean variable which can be true or false based on some criterion. Depending on if the decision tree is a classifier or a regressor, a leaf is a class label or a predicted value of the target variable. In the case of Regression Trees, which is the type of decision tree that is used in this project, a leaf is assigned the mean value of the data points in that leaf. For a new set of features, one would travel down the tree until a leaf is reached and whatever value this leaf has will be assigned to the new prediction.

Decision trees are an example of a greedy algorithm [14], which means they are (usually) grown top-down and make locally optimal choices at each node. Thus, they are not guaranteed to converge at a global optimum. There are ways of improving their chances of finding a global optimum, which will be discussed later, but for now, let's focus on how locally optimal choices are made.

The most commonly used methods make splits based on some information quality criterion [15]. The four most common ones are *Gini Impurity*, *Information Gain*, *Variance Reduction* and *Mean Squared Error* (MSE). MSE is the most commonly used method in the case of regression trees, and this will be described in detail later. In classifier trees, the Gini Impurity is often used. For the sake of the iris data set example, let's briefly study this measure of split quality.

The Gini Impurity is defined as the probability of a randomly chosen element from a set being incorrectly labeled if it were randomly selected according to the distribution of labels in the set [15], i.e for a set of m items, suppose $i = \{1, 2, \dots, m\}$ and let f_i be the fraction of items labeled i , then the Gini impurity is mathematically defined as in equation 2.1. Note that this function reaches its minimum

when all items in the set have the same label.

$$I_G = \sum_{i=1}^m f_i(1 - f_i) \quad (2.1)$$

In figure 2.8, the process of splitting and evaluating in a decision tree node is shown graphically. Note that while the figure illustrates a single split at a certain location, all possible splits on all available features (often with replacement) would be considered in each node and the resulting partitioning of the target variable evaluated using the Gini Impurity. The split that yielded the lowest Gini Impurity is chosen for the node.

The tree is recursively grown in this manner until some stopping criterion is met. This criterion might be the height of the tree (the number of node levels) or the number of data points in each leaf. Since decision trees are sensitive to over-fitting [14], it is important to choose the stopping criterion correctly. The tree could potentially be grown until there is only one data point in each leaf and then the tree would be heavily over-fitted.

In summary, the steps in decision tree growing can be summarised by the following steps:

1. Begin in the root of the tree. Loop through all features and split the target variable at all possible points. Calculate the resulting Gini impurity. Choose the feature and split that yielded the lowest Gini impurity.
2. For each new node, repeat 1 and continue down the new branches in a recursive manner.
3. If the number of data points in a node is lower than some threshold, then make the previous node in that branch a leaf.
4. When all branches have leaves, stop.

Let's now return to Fisher's iris data set and see how a decision tree works in this example. Recall that there are four different features in the iris data set: the petal and sepal width and length. There are 150 measurements of these features available along with the same number of recordings of the iris species. The iris species is the target variable and consists of 3 class labels: *iris setosa*, *iris virginica* and *iris versicolor*. Typically, class labels are transformed into an integer and in this case the classes are 0, 1 and 2 for setosa, virginica and versicolor, respectively. Training a decision tree on the iris data set yields the tree structure seen in figure 2.9.

It can readily be seen that the most important feature is the petal length. This feature is used to distinguish setosa from virginica and versicolor. As this split completely separates setosa from the other species, there is only one label present

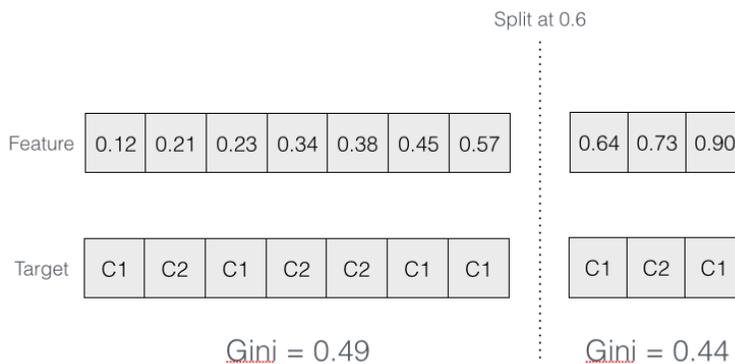


Figure 2.8: Splitting a node based on the Gini impurity.

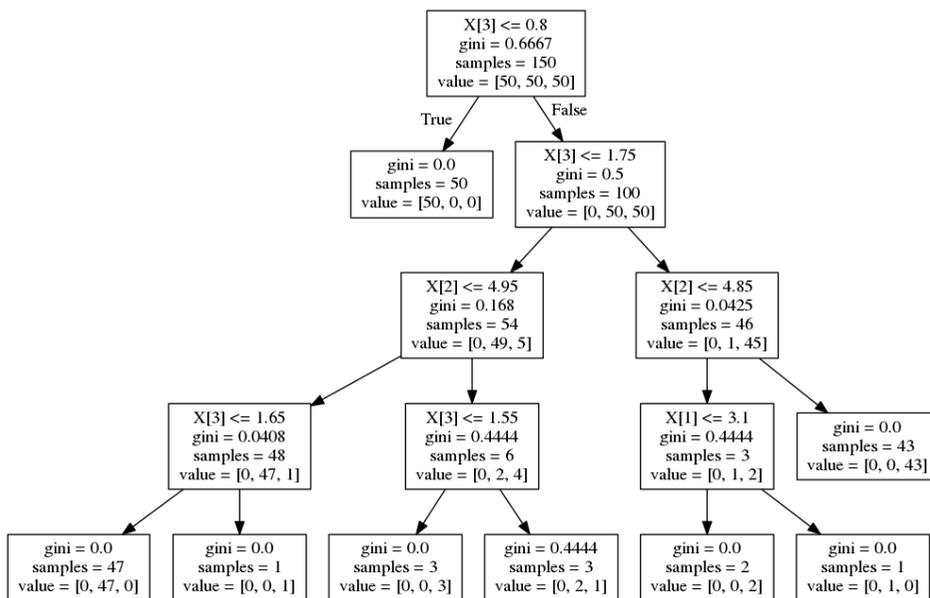


Figure 2.9: A decision tree trained on the iris dataset. $x[1]$ is sepal length, $x[2]$ is sepal width, $x[3]$ is petal length and $x[4]$ is petal width. The *value* variable indicates how many instances of setosa, virginica and versicolor, respectively, there are in the node.

in the first node of the left branch. Thus, the Gini Impurity is zero, no additional splits are required and this node becomes a leaf. In the right branch however,

additional splits are needed to distinguish *virginica* and *versicolor*. Interestingly, one can see that petal length is used to make splits more than once in this branch, indicating that there are indeed non-linear relationships between the features in the iris data set.

It can further be seen that there is one leaf in the tree that does not have 0 Gini impurity, i.e there is more than one class in the leaf. This implies that while the four features tell us quite a lot about the typing of the flower, they do not completely separate the three species. In the particular case of iris flowers, it appears that the petal width does not provide any unique information about the species of flower, as it is not used in the tree. This demonstrates another limitation of machine learning and decision-making in general: it is completely limited by the information available in the data. In this scenario, there might be some other unknown feature which would provide more information on the species of the flower and further increase the performance of the tree. It is further worth mentioning that while this tree seems to separate the species very well, it is likely that the depth of the tree has made it over-fitted. Thus, for a new set of feature measurements, the tree might classify the species incorrectly.

2.2.2 Regression trees

Regression trees are identical to classification trees in all aspects except of how splits are performed [14]. As previously mentioned, the MSE is often used to determine the quality of the split in a regression tree. As the leaves of the tree will no longer be a set of class labels, but rather a set of continuous values, the MSE gives a more suitable measure of quality. It is defined in equation 2.2, where μ is the expected value (mean) of m data points in a set, and y_i , $i = \{1, 2, \dots, m\}$ are the individual measurements.

$$MSE = \frac{1}{m} \sum_{i=1}^m (\mu - y_i)^2 \quad (2.2)$$

It can readily be understood that minimising the MSE will encourage splits where the data points in the resulting subsets are more homogeneous.

Just as in the case of the classification tree, splitting is recursively performed until a stopping criterion is met. When a new set of feature measurements is to be evaluated, the target is predicted as the mean value of the data points of the leaf one ends up in for that set of features.

2.2.3 From regression trees to random forests

There are several advantages of using decision trees in prediction scenarios, but those that are most important for this project are the following:

- The white box model [14] of decision trees gives us information on the predictive power of the different features.
- Decision trees are able to handle both numerical and categorical data (recall that questions in A New Universe generate both continuous and binary values).
- Decision trees perform well with large data sets and do not require large computational power.

There are also some drawbacks to using single decision trees. While they have low bias, they suffer from high variance [14]. This is attributed to the greediness of the method and the fact that decision trees are very easily over-fitted. Thus, single decision trees are generally very unreliable and can often begin to model noise instead of the underlying structure of the data. In 2001, Leo Breiman at the University of California introduced the concept of so-called Random Forests [16], an ensemble method which deals with the high variance of decision trees and thus greatly increases the performance of the method.

This method uses bootstrap aggregating [16] and is fairly simple. The key concept is to train several trees on different subsets of the data, and then take the mean of the trees' predictions (or let the trees cast votes in the case of tree classifiers). While the variance of a single tree is very high, taking the mean of several trees will lower the variance, while only increasing the bias by a small amount. For this to be true, the trees must be uncorrelated, something which is achieved by selecting subsets of training data at complete random (hence the name random forest). The number of trees that should be used varies from problem to problem, but the training and test errors tend to level off after some number of trees have been fit [16]. As for the number of features used in the subset for each tree, for a total of p features, the inventor of the Random Forest method recommends \sqrt{p} for classifiers and $p/3$ for regressors [16].

2.3 Genetic Algorithms

According to the Theory of Evolution, life has reached its current state through *natural selection* - the engine that drives evolution forward [17]. As an organism changes with time, it obtains certain characteristic that are abnormal. These characteristics could prove to be either beneficial or harmful. The strongest organisms survive and become the "new normal" while ones that could not adapt die out. According to the theory, organisms will continue to evolve and change until they find their optimal fit in an environment.

It has been showed that methods similar to natural selection can be used to find solutions to optimisation and search problems in computation [18] [19]. An

evolutionary algorithm uses mechanisms that are inspired by biological processes, such as reproduction, mutation, recombination and selection. Possible solutions to an optimisation problem are represented as individuals in a population, where evaluation through a so-called *fitness function* will decide the quality of the specific solution.

A genetic algorithm (GA) is a kind of evolutionary algorithm that simulates natural selection by computing the fitness amongst individuals over consecutive generations to solve an optimization problem. Each generation consists of a population of individuals that are represented by a combination of variables which together create a solution. The variable values for each individual will change through mating and mutation. Parents are chosen, from the strongest individuals, to mate and pass on their variable information to their offspring, an operation that opens up new solutions to the optimisation problem. As generations pass, individuals will evolve and adapt to their environment, i.e. find the optimal solution for the problem.

To further explain the concept of genetic algorithms, an example problem will be presented and the process of creating and running a genetic algorithm on it will be shown, step by step. An overview of how a GA is run can be viewed in figure 2.10.

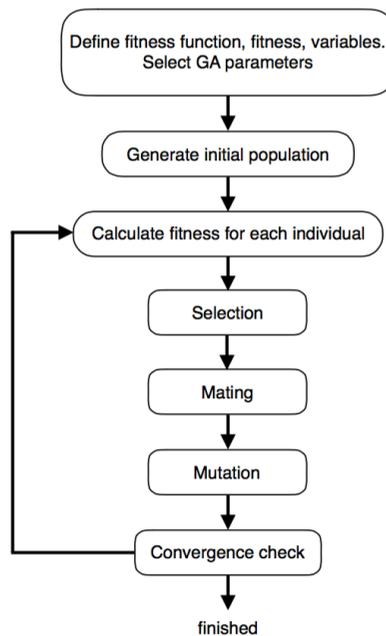


Figure 2.10: Flowchart of a GA.

Now, imagine a mountain range containing several peaks of similar height. In this example, the mountain range is defined by a three-dimensional function with a few local maxima. The function, defined by equation 2.3, is a standard MatLab example function called *peaks* [20] and is illustrated in 2.12.

$$\begin{aligned}
 g(x, y) = & 3 * (1 - x)^2 * \exp(-(x^2) - (y + 1)^2) - \\
 & 10 * (x/5 - x^3 - y^5) * \exp(-x^2 - y^2) - \\
 & 1/3 * \exp(-(x + 1)^2 - y^2)
 \end{aligned}
 \tag{2.3}$$

The goal for the GA in this particular case is to search the coordinates and locate the global maximum value of the function described by equation 2.3. Note that GA's are commonly used on more complex problems, particularly high-dimensional optimisation problems, but a fairly simple problem is used here for demonstration.

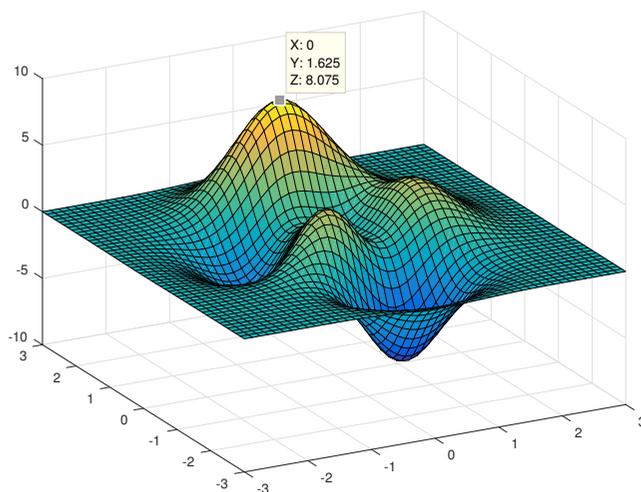


Figure 2.11: A three-dimensional plot of the function described by equation 2.3.

The maximum value could easily be found by searching through every coordinate and comparing the respective value. The draw-back to using this approach, however, is that it would be time consuming (even more so for more complex problems). Instead, the GA will search through the area from a number of random start points and successively improve the solutions until, hopefully, the maximum

value is found. For this example, one could consider other, more conventional optimization techniques, but many of these have difficulties finding the global maximum in more complex problems, especially if there is a large number of local maxima [21].

2.3.1 The individual

The individual is defined as an array of variable values - *genes* - which are to be optimized. Each individual represents one possible solution. An individual can be compared to nature's own counterpart: a chromosome with genes as the variable values. The number of genes in an individual is chosen to match the problem at hand. If the individual has N_{var} genes given by $p_1, p_2 \dots, p_{N_{var}}$ it will be written as a vector with N_{var} elements,

$$Individual = [p_1, p_2, p_3, \dots, p_{N_{var}}] \quad ,$$

where the variables are continuous and restricted by a lower and an upper limit,

$$a < p_i < b \quad .$$

Each individual is assigned a score in the form of fitness, where the individual with the highest fitness is most apt to survive and allowed to pass on its genetic material. The fitness is evaluated through the fitness function, which is an objective function that is used to determine how close a given solution (i.e individual) is to the pre-set aims. The fitness score is calculated directly from the individual's gene values 3.2.

$$score = f(individual) = f([p_1, p_2, p_3, \dots, p_{N_{var}}]) \quad , \quad (2.4)$$

where $f(\cdot)$ is some fitness function.

In this example, each individual C_i will consist of two genes ($N_{var} = 2$); one representing the x-coordinate and the other representing the y-coordinate:

$$C_i = [x, y] \quad ,$$

with the constraints $-3 \leq x \leq 3$ and $-3 \leq y \leq 3$.

In this case, fitness is easy to understand and describe: one simply wishes to find the value of the mountain range function at the coordinates described by the individual (2.5).

$$f(C_i) = g(x, y) \quad (2.5)$$

2.3.2 Basic setup of the genetic algorithm

To set up a GA, an initial population of N_{pop} individuals is defined. The population can be represented by a list of rows where each row is an individual that consists of a $1 \times N_{var}$ vector of genes. Each value in the rows is generated with a random generator, to spread the individuals evenly over the search space.

A population of $N_{pop} = 8$ individuals (table 2.3), with two genes each, are generated in the example. The individuals' locations in the mountain range is plotted in figure 2.12.

| Individual | Genes | | Fitness |
|------------|--------|--------|---------|
| | x | y | |
| 1 | -0.125 | 0 | 1.447 |
| 2 | -1.0 | 2.25 | 1.322 |
| 3 | 2.750 | -0.375 | 0.095 |
| 4 | 2.875 | -2.875 | 0 |
| 5 | 1 | -2 | -2.102 |
| 6 | 1 | 0.5 | 2.376 |
| 7 | 0.625 | -0.875 | -0.971 |
| 8 | -3 | 1.875 | 0 |

Table 2.3: Initial population of 8 random individuals with their corresponding fitness score.

2.3.3 Natural selection and evaluation of fitness

The concept of survival of the fittest is applied in the GA by selecting only the most fit individuals to mate and evolve. The population of individuals are ranked from highest to lowest fitness and only the strongest individuals are selected to mate, while the rest are discarded. A selection rate X_{rate} is chosen to decide how many individuals will survive and be added to the mating pool N_{keep} , equation 2.6.

$$N_{keep} = X_{rate} * N_{pop} \quad (2.6)$$

In the example, a selection rate $X_{rate} = 0.5$ is chosen for the population of individuals, i.e, only the top half will survive to the next generation. The surviving individuals can be seen in table 2.4.

There are several other ways to choose the selection rate which all have their advantages and disadvantages [22], but in this simple example the selection of the 50 % strongest individuals will suffice.

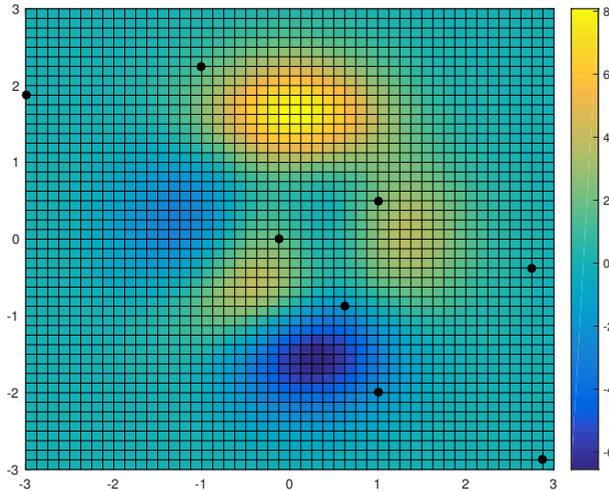


Figure 2.12: A heat-map with the initial population from table 2.3 shown as black dots. Colors indicate the value of the fitness function, or height of the peaks.

| Individual | Genes | | Fitness |
|------------|--------|--------|---------|
| | x | y | |
| 6 | 1 | 0.5 | 2.376 |
| 1 | -0.125 | 0 | 1.447 |
| 2 | -1.0 | 2.25 | 1.322 |
| 3 | 2.750 | -0.375 | 0.095 |

Table 2.4: The 50 % strongest individuals.

2.3.4 Selection

Individuals are selected to mate and create offspring. Potential parents are selected from the mating pool and the goal of the selection process is to select the strongest individuals. The parents are usually not selected from top to bottom in the mating pool, but are rather chosen in a process with some degree of randomness. To explain why the random element is important, a comparison can be made to the animal kingdom; the strongest individuals are dominant and most apt to procreate but it is far from sure that they will find the strongest counterpart in the opposite sex. This uncertainty and randomness is part of nature's

way of mixing genetic information which opens up for new combinations in their offspring's genetic material. This is a foundation-stone in evolution.

There are several different existing selection methods and there are no actual standard for how they should be constructed. Methods vary from choosing parents randomly from the surviving individuals, to more complex methods which mimics nature better. Some known examples of selection are *roulette wheel*, *stochastic universal sampling* and *tournament selection* [23]. The choice of method will affect the result of the GA so it should be chosen with care. As tournament selection is used in the example problem, let us briefly introduce its concept.

Tournament selection - A small subset of individuals (usually two or three) are randomly picked from the mating pool, where the individual with the highest fitness in the selected subset becomes a parent. This continues until a desired number of parents have been chosen.

An example of how parents are selected can be seen in figure 2.5. Here, individual # 6 was chosen as a parent two times - something which may happen as selection is performed with replacement.

2.3.5 Mating

The selected parents pass on their genetic information to the offspring in the GA's mating stage. There are a variety of mating schemes to choose from where, in most cases, two parents mate to create two offspring. A common method is the *single point crossover* [24], where a crossover point along the individual is randomly selected. This point indicates a split point in both parents' set of genes. $parent_1$ will pass on the genes to the left of the crossover point to $offspring_1$ while $parent_2$ passes its corresponding genes to $offspring_2$. In similar fashion, the genes to the right of the crossover point of $parent_1$ goes to $offspring_2$ and $parent_2$ passes its genes to $offspring_1$. An illustration of this mix of genes is shown in figure 2.13. The offspring created is a new individual with the same basic construction as its parents but with a new combination of genes that is a mixture from both parents.

A problem with the point crossover method in this example is that no new information is introduced, only values that have been randomly introduced in the initial population exist. This information can be combined in a large number of ways, but there might still be some genes which are not represented, if they were not initially introduced to the population at seeding. A blending method (seen in equation 2.7) solves this problem by combining the variable values from the two parents into new variable values in the offspring [25]. One new offspring gene, p_{new} , is generated from a combination of the two corresponding gene values from the parents. This procedure is used to create enough genes to construct a new offspring, according to

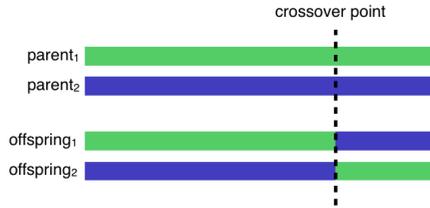


Figure 2.13: An illustration of single point crossover. The colored bars represents the genes of an individual.

$$p_{new} = \beta * p_{m,n} + (1 - \beta) * p_{f,n}, \quad (2.7)$$

where β is a random number on the interval $[a, b]$, p_{mn} is n :th variable in the mother individual and p_{fn} is n :th variable in the father individual.

The number of parents chosen and offspring created is often connected to the selection rate, X_{rate} . After parents have been selected and they have mated to create offspring, the population is pruned back to the initial number of individuals, N_{pop} . Thus, the population size between generations is constant.

The Beta blending method is applied to the example problem and the resulting offspring can be viewed in figure 2.5.

| Individual | Individual | x | y |
|------------|------------------------------|--------|--------|
| 1 | <i>mother₁</i> | -1.0 | 0 |
| 6 | <i>father₁</i> | 1 | 0.5 |
| new | <i>offspring₁</i> | 2.5 | -0.5 |
| new | <i>offspring₂</i> | -1.5 | -2 |
| 6 | <i>mother₂</i> | 1 | 0.5 |
| 2 | <i>father₂</i> | -1.0 | 2.250 |
| new | <i>offspring₃</i> | -2.250 | -2.250 |
| new | <i>offspring₄</i> | 0.375 | 2.250 |

Table 2.5: Parents selected, and the offspring created, with their corresponding gene values.

2.3.6 Mutation

Suppose that the GA finds a solution after just a few generations, this would generally be seen as a good sign. It can imply two things: either the problem has

a very simple solution, or the solution was actually a suboptimal solution - a local optima instead of a global one. In practice, the latter scenario is often the case. In order to avoid this scenario, mutation is added to the algorithm. This is another way for the GA to explore the solution space in which it can introduce new traits that did not exist in the original population.

Through mutation individuals are randomly altered, a process where a gene is slightly modified. Modifications are done through single point mutations and are applied - usually with a low probability - to a randomly chosen gene in the individual. Parameters that need to be set are mutation rate (M_r) which decides the probability of a gene being mutated and mutation step (M_s) which decides the size of the change. By increasing the number of mutations, the algorithm's ability to search for solutions outside the genes represented by the current individuals increase. However, if the mutation rate is too high it will take time for the GA to converge towards any solution at all, as the process will begin to mimic a random walk.

To make sure that the best individuals stay intact, an elite rate (M_e) may be selected. This rate is a percentage of the population which cannot be mutated.

In the example problem, a mutation rate, $M_r = 0.05$, is chosen, which means that 5 % of all genes in the population will be mutated, except for the elite individuals. A number of genes, corresponding to 5 %, are randomly selected from the whole population. If a gene is chosen for mutation, the current value will simply be replaced by a new random value within the boundaries:

$$1.64 \rightarrow -2.41$$

2.3.7 Evolution and convergence

The number of generations that a GA is allowed to run usually depends on whether an acceptable solution has been found or a certain number of generations has passed. In a generation, all the previously described steps are performed, according to figure 2.10. After a number of generations, all individuals will usually converge towards a specific gene setup, as the strongest individual dominate the population with its genetic material. At this point the process is usually terminated.

In figure 2.14, the result of applying a GA to the example problem of the mountain range can be seen. In this simple problem, the solutions in the figure are found after only 20 generations. Some individuals still linger on a local optimum, but the strongest individuals are close to the global optimum. Given additional generations, the sub-optimal solutions would converge towards the optimal solution. In the end, all individuals in the population will be identical.

A common way to keep track of the GA's performance is in the form of mean and maximum fitness. This makes it possible to see the GA's convergence towards possible solutions and opens up for another way to follow the improvement of the

solutions. When the mean, or maximum, score does not improve enough from one generation to the next, it means that a solution has been found, optimal or suboptimal, and the GA should be stopped. GAs are another example of a greedy algorithm, and local optima are often reached. The parameters described throughout this section should thus be chosen to give the GA the best possible chance to find an global optima.

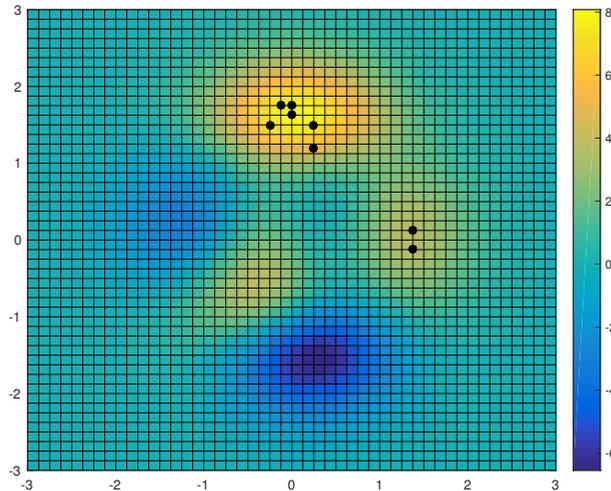


Figure 2.14: A heat-map with the final population shown as black dots. Colors indicate the value of the fitness function - or the height of the mountain peaks.

2.4 The Genetic Algorithm - Random Forest Hybrid

A fairly common method for improving the performance of random forests is to limit the number of features the method can use for prediction. It is not uncommon for random forests to yield poor results if there are too many features to choose from [14]. This issue is once again related to over-fitting, something which we have previously noted is a returning issue with random forests. Further, if one is interested in which specific features are optimal in predicting the target, it may be difficult to identify these optimal features among a very large number of total features used in the random forest - especially if there are many different sets of features that yield similar results.

There exist a number of methods that can reduce the dimensionality of the data, for example Single Value Decomposition (SVD) [26] or Principal Component Analysis (PCA) [27], but an approach that is becoming more popular because of its ability to quickly search high-dimensional spaces for optimal solutions is to use genetic algorithms for feature selection.

This method, referred to as *GenForest* by Hansen et. al [28], is fairly common in prediction scenarios where the relevance of individual features is unknown. It has been shown to yield improved results while only using a small fraction of the available features [28] [29] [30]. These findings confirm the observations made by the creators of the random forest algorithm - that random forests seem to work best on a small number of informative features [14].

In the Random Forest - Genetic Algorithm prediction method (referred to as RFGA from now on), individuals in the population are represented by binary arrays of the same length as the number of features. A 1 in this array indicates that the corresponding feature should be used by the random forest, while a 0 indicates that it should not be used. The fitness of GA is usually some measure of the random forest prediction quality for the set of active features, determined by the individual's genome, such as the RMSE of prediction performed on a test set. This is combined with some sort of fitness measure that encourages individuals with fewer active features to be produced. To seed the population, a random set of genes in each individual are activated. The GA is evolved in the way previously described, with some changes to how mating and mutation is applied. The one-point crossover method previously described is often used, as this has been found to be the most effective method [31]. How the method differs from the GA example previously described is discussed in greater detail in the Method section.

Chapter 3

Method

3.1 Goals and Requirements

3.1.1 Main goals

Participation and data quality are the main incentives behind this project, and are directly connected to the project's two main goals.

Study the possibility of using machine learning techniques on self-report data to:

1. Improve the quality of the user experience in A New Universe by personalising the interaction between the app and the user.
2. Improve the quality of the collected data.

While these goals may seem straight-forward enough, there are many implications that must be considered. We will begin by studying the second of the two goals in greater detail.

3.1.2 Improving data quality

How one would approach the challenge of improving data quality greatly depends on the definition of high and low quality in the particular case. It has previously been mentioned that in this project, the data consists solely of answers to various questions on mental health. Thus, in this project, an individual answer could be considered to be of high quality if:

- It is truthful
- Thought has been put into it
- The question was relevant to the individual

The quality of the answer is reduced if any of these three requirements are not met. Obviously an answer does not yield any relevant information about the individual if it is not truthful, but even if a user may be truthful when answering questions, if he or she hurries through the questions without interest, the answers may not be entirely accurate. If a user is both truthful and thoughtful when answering a question, the quality of the information gained from that answer is still lowered if the question was not relevant to the specific user to begin with. The aspect of relevance is something we will return to later, and define in greater detail.

These three factors that affect information quality is something over which we have little control (but not no control, as we shall see later). Therefore, from a data quality perspective it is more meaningful to consider the composition of the data.

Ideally, one would of course like to have high data coverage both among the variables and in time, i.e all questions answered by all users in all sessions. In other data mining scenarios, achieving such high coverage may very well be possible. In this case, however, it is not. The perhaps most important characteristic of A New Universe is that the flow of data is completely dependent on the active and willing participation of the user. Obviously, posing 160 questions to a user in each and every session is not reasonable - it would simply put too much strain on the user, increasing the likelihood of that user opting out of the app (something that would affect data collection negatively).

If one cannot have it all, it may be tempting to instead focus on a few variables and gather as much data as possible on these. This is essentially a sound idea, and we will soon come back to it, but for now we will only point out that the outcome of such an approach greatly depends on which questions one chooses to focus on.

Another approach is to do the opposite of the previous example and gather a little data on each variable, but instead study a large number of variables. This is essentially how A New Universe has previously functioned. While such a variety in the questions could perhaps have a positive impact on the user experience, the main issue with this approach is that a user's answers to a question have fairly high variance and are not necessarily stationary. Therefore, a few random, temporally scattered samples from a large number of questions does not tell us much about the user. This essentially means that the quality of the data collected with this method is poor.

Ultimately, what one ideally would like to achieve - while being realistic - is data coverage on as many variables as possible, while maximising the number of samples available in each variable. In a defined set of subsequent sessions, this trade-off should favour few questions and many samples. In the long-term, however, which questions one focuses on should be allowed to vary.

Further, the data collection method should be designed to focus on questions whose answers are relevant in describing the specific user. This would ensure that even though data has not been gathered on all variables, the data that is available is of higher relevance and thereby of higher quality. This criterion of relevance is directly connected to the first main goal of the project.

3.1.3 Meaningfulness

In order to create a positive user experience in A New Universe, understanding the incentives behind a user's participation is key. While these incentives surely differ from person to person, it is reasonable to assume that an individual first downloaded the app because he or she suffers from some mental health issue. Some users may want to understand these issues better, while others may simply seek the relief that can come from sharing one's thoughts. Yet others may be seeking tools to better cope with their situation. These different types of users all have different expectations on the app - expectations that are further affected by the severity of the users' issues. For those with mild or passing mental health issues, the app might only be a fun experiment. Those with more severe issues may invest much more in the app. Regardless of what the expectations on A New Universe might be, it is important that the app feels meaningful to the user.

In many forms of psychological therapy, the result of a treatment is dependent on the patient's willingness to participate. A study published in *Journal of Clinical Psychology* in 2009 found that patients who had received their preferred treatment were half as likely to drop out [32]. The importance of willingness does not only apply to the initiation of therapy, but to the patient's attitude throughout the treatment. The patient's wish to see improvement, and his or her belief that the treatment may help have been shown to contribute to a positive treatment outcome [32]. We will summarise these factors of willingness, motivation and conviction in the more general concept of *meaningfulness*.

In the case of A New Universe, where treatment is not the main function, meaningfulness is still an important concept. In a similar fashion to a patient going through therapy, a user in A New Universe must feel that he or she has something to gain by using the app, otherwise the risk of that user losing interest is high. Regardless of if an individual downloaded the app willingly, or under the instruction of a therapist, he or she should feel that the interaction with the app is meaningful. If this can be achieved, one may have some control over the three previously mentioned factors that define a high quality answer. How then,

does one create meaningfulness? There are, of course, many ways to approach the problem, but in this project we will focus on the selection of relevant questions.

As was previously mentioned, the 160 questions in A New Universe are divided into 13 categories, some of which are rather specific and may not be (or appear to be) relevant to all users. For example, questions concerning alcohol habits are not relevant to someone who does not regularly consume alcohol, just as questions concerning sleep issues are not relevant to someone who has never had such issues. However, if the person is feeling left-out or bullied at school or work, questions from the “Relationships” category might be highly relevant. Ideally, the app should recognize this and explore that area more thoroughly, in a similar fashion to how a therapist would evaluate a patient through dialogue and attempt to identify his or her problem areas.

There is a basic aspect of the human psyche that makes the detection of relevant questions difficult in an app like A New Universe: it is not constant. For example, an individual’s alcohol and sleep habits may change as time passes, and if these question categories become more relevant a personalisation algorithm must be able to detect the change. Conversely, questions whose answers do not appear to change significantly over time should not be posed more often than what is needed to maintain a good understanding of the individual’s situation.

An important factor that will heavily affect the algorithm’s ability to address these issues is data sparsity. How does one evaluate the relevance of a question or category for a specific user when there is so little data available on that user? Simply posing identical questions over a period of time to evaluate them is not a realistic approach, as in the cases where the questions turn out to be irrelevant, this would negatively affect meaningfulness. It seems that we are faced by a kind of Catch 22: we wish to find questions that are meaningful to the user, but to find them we must gather data on the questions, something we cannot do without negatively impacting the meaningfulness. For new users, on whom we have no data at all, this issue becomes even greater. As the drop-out rate on new users has historically been fairly high in A New Universe, immediately sparking a new user’s interest is important.

Another factor that makes finding relevant questions difficult is the fact that the vision of the A New Universe project is to not put emphasis on the answers to individual questions. The idea is to not consider whether the question has a positive or negative connotation, i.e if it is formulated as “How good are you feeling today?” or “How bad are you feeling today?”, but only find the absolute value of the potential correlation. While this would remove the need for defining how every individual answer should be interpreted, it obviously presents some other issues.

3.1.4 Key requirements

We have now reflected on the main goals of this project and gained some important insights that allows us to describe the greatest challenges of the project in a few core points:

- From a data coverage perspective, one would ideally like to have answers to all questions in all sessions. This is not possible, however, due to the negative impact this approach would have on the user experience, and in extension, long-term data collection.
- The best compromise is to instead focus on a smaller set of questions at a time, and gather as much data as possible on these. In order to avoid sacrificing the user experience and to ensure that the gathered data is meaningful, focus should be put on questions that are relevant to the specific user.
- Since there is a high variance in the answers to questions, and because the answers are not necessarily stationary, finding relevant questions is problematic. One would ideally like to quickly identify which questions are relevant to a specific user, but cannot easily do this without using a trial-and-error approach to testing questions - a method that would negatively impact the user experience.
- For completely new users, no data whatsoever is available from which to evaluate question relevance. Thus, finding optimal questions for these user's is difficult.

These challenges now takes us to the key requirements of the presented algorithm.

For a question selection algorithm in A New Universe to be effective under the constraints of the main goals, it must be able to

From a personalisation perspective:

1. For a specific user, create a dynamic question profile by
 - (a) Measuring the relevance of the different questions with some method.
 - (b) Measuring the stability of the answers to these questions to decide how often they should be posed.
 - (c) Adapting to changes in the answers to questions to detect events in the user's life.
 - (d) Circumvent issues with data sparsity in a user's answers.
 - (e) Handle new users, on whom there is no data available.

From a data coverage perspective:

2. Maintain a high level of data coverage, despite the fact that a personalisation algorithm will narrow the scope of data collection.
3. Allow for questions to be manually selected and posed to users*.

**This point is not something which we have previously discussed, but is a feature that an effective algorithm should include to allow for hypothesis testing. In a research scenario, such control over which questions should be posed to the users is valuable. Further, if a new question is added to the app, one might wish to quickly test its relevance.*

The formulation of these key requirements is an important part of this project and we will return to them many times throughout this report, as they will be the foundation upon which we build the arguments for the advantages of the presented algorithm. It is worth mentioning that evaluating the impact of the question selection method in terms of improved user experience is difficult in this case, as it would require studying the response of participants over a longer period of time. As the test group is fairly small and the project limited in time and resources, this is not realistic. Instead, focus is put on evaluating the behaviour of the algorithm and determining if it can be used to differentiate between different questions in terms of meaningfulness. This project should be seen as a pilot study that is a first step on the road to intelligent mental health data collection systems.

3.2 A Brief Summary of the Presented Algorithm

In the set of mandatory questions in A New Universe the simple question “How are you feeling right now?” is included. This question is the base upon which the presented algorithm is built, and is used as a benchmark for finding other relevant questions. The reasoning behind this choice of benchmark is that out of all the questions in A New Universe, this question targets the user’s well-being in the most direct fashion and is the best available measure of the user’s current psychological state. Further, as it is a mandatory question, there is data available for all completed sessions.

Let’s immediately introduce the hypothesis upon which the presented algorithm is built:

Hypothesis. *If there exists a relationship between the answers to a specific question and the answers to the question “How are you feeling right now?”, then that question is relevant to the user.*

This hypothesis does not hold in all scenarios, of course, but is a reasonable assumption to make if one considers the fact that most users in A New Universe have turned to the app because they want to learn more about what aspects of their lives affects their mental well-being. As has previously been discussed, areas that affect the user’s well-being should be identified and explored further, something which will be possible by making use of the benchmark question.

The potential relationship between answers to a specific question and the answers to the benchmark question is measured in two principal ways in the presented algorithm. These methods will now be presented, together with some other important components of the algorithm. Recall the key requirements described in subsection 3.1.4 on page 41. How these requirements are fulfilled in the presented algorithm will now be briefly summarised.

We have previously mentioned that historically, it has been fairly common that participants stop using A New Universe after only one or a few sessions. While the reasons behind these early drop-outs could be many, this fact further stresses the importance of quickly attracting the interest of new users. As there is no information available on a first-time user, the best available solution when it comes to finding optimal questions to pose is to study the already collected data and find questions that are relevant on a group level. While such an approach might not find questions that are relevant on an individual level, it will be able to identify questions that are generally relevant.

To find optimal questions to pose to new users, we suggest an RFGA prediction method, trained on question data from all users, to find questions that are related (linearly, or non-linearly) to the benchmark question.

For recurrent users, we suggest an ad-hoc *parameter-based question evaluation* method (referred to as PBQE from now on) for evaluating the user-specific relevance and stability of different questions (addressing the first few points in the list of requirements). The main reasoning behind using this method instead of more complex machine learning techniques is that the latter often fall short when there is too little data available. Thus, a simpler parameter-based method is a more reasonable choice of method for detecting changes in individual users’ answers and test the relevance of infrequently posed questions. In this method, a simple event-detection method that can adapt the question selection scheme to changes in the user’s answers is also incorporated.

In order to handle the issue of data coverage (points 4 and 5 on the list of requirements in section 3.1.4), we suggest a simple method that on a daily basis selects a set of questions and poses them to all users in A New Universe. This method also allows for questions to be manually selected and posed to users.

3.3 Data Pre-processing

3.3.1 In the *Random Forest - Genetic Algorithm* prediction method

As previously mentioned, type 3 questions in A New Universe produces answers that are represented by a binary string of zeros and ones. To handle these answers when predicting, they are split into several type 0 questions, with answers that are either a zero or a one depending on if the particular alternative was checked or not. If a type 3 question has seven alternatives then the answer is split into seven type 0 answers. The same splitting is performed on type 4 questions. This procedure effectively increases the number of total features, in the form of questions and sub-alternatives to questions, from 160 to 546.

Some type 3 and one type 4 question have been changed during the testing of A New Universe, so that these questions have had different numbers of alternatives at different time-points. This of course means that the data from these questions is corrupted. To handle this, only answers that have been collected from the question in its current form is considered in the algorithm.

Further, older versions of type 1 questions have not been normalised so that the answer is a decimal number between zero and one. Data from these questions is correctly normalised before it is used in the algorithm.

The raw data that is used by the random forest is stored in matrix format. The benchmark question data is stored in an array of size $N_{sessions} \times 1$, where $N_{sessions}$ is the total number of sessions completed by all users. As the benchmark question is a mandatory question, there are no missing values. The features, i.e the answers to all questions except the benchmark question, is represented by a matrix of size $N_{sessions} \times N_{features}$. $N_{features}$ is as previously mentioned 546, and the number of rows ($N_{sessions}$) is the same as in the benchmark array. The origin of the integer $N_{sessions}$ may be difficult to immediately understand, so let's look at an example.

If User A has completed two sessions, and answered 10 unique questions in session one, and 10 unique questions in session 2, then that would yield a 2 by 546 matrix. In each of the matrix's two rows, there would be 536 *NaN* values, and 10 actual answers. If User B has completed one session and answered 15 questions, then that would yield a 1 by 546 matrix (again, with mostly *NaN* values). The complete matrix on which the random forest is trained would be a 3 by 546 matrix, as the two users' answers are vertically stacked on each other. Note that in this matrix, a type 3 or type 4 questions are represented by more than one column, and each column corresponds to one of the answer alternatives to the question.

As random forests do not generally handle missing data, the *NaN* values are imputed using the median of the specific column.

3.3.2 In the *Parameter-based Question Selection* method

In the same manner as in the RFGA method, type 3 and type 4 questions are split into several type 0 questions and parameters are calculated for each sub-answer. This is explained in greater detail in section 3.5.1.

3.4 Selecting Optimal Questions for New Users

In this section, we will systematically go through the different parts of the RFGA prediction method, focusing on the parameter values that are used in the final version of the presented algorithm. A variety of different parameter values were examined and the impact of these different values on the outcome of the algorithm are presented in detail in the *Results* and *Discussion* sections of this report.

3.4.1 Basic setup of the genetic algorithm

Individuals in the RFGA hybrid are represented by a list of genes that are either zeros or ones, with a length of 546 (the number of total features). As mentioned in the background, a one indicates that the specific feature should be available for the random forest to use for training and testing, while a zero indicates that it is inactive and should not be used in the prediction. To initialise the GA, 100 individuals with all 546 genes set to zero are created. A random set of genes are then activated in each individual, with each gene having a certain probability of being activated.

3.4.2 Fitness

In every generation of the algorithm, each individual is evaluated based on its fitness. This fitness is defined as the sum of two sub-fitnesses:

1. The root mean square error (RMSE) of the random forest's prediction on a test partition of the data.
2. The relative number of active features.

To evaluate the fitness of a specific individual, the instructions in the form of the individual's zeros and ones is used to create a new data matrix on which a random forest is trained. This new matrix will have the dimensions $N_{sessions} \times N_{active\ features}$ - i.e, only the number of columns is different from the original data matrix (since only a small number of features are now active). In figure 3.1, an example of how features are selected is shown.

This new matrix is then randomly split into a training and a test partition, with 70 % of the data being used for training and 30 % for testing, i.e, for $N_{sessions}$

individuals that minimise this function.

3.4.3 Evolution

For the exchange of genetic material, the so-called $\mu + \lambda$ mating scheme is used [33]. In this scheme, the tournament selection method mentioned in section 2.3.4 is used to select a group of individuals that are allowed to mate. The one-point crossover method, which is as previously mentioned the standard for binary individuals, is used to generate λ offspring from these parents. Mutation is then applied in the form of bit-flipping, with an individual gene having a certain probability of being flipped (0 flipped to 1 or 1 flipped to 0). This procedure prevents the algorithm from getting stuck in a local optima early on in the optimisation process. Several different mutation rates were examined and a 10 % probability of mutation was chosen for the presented algorithm. Each gene in the chosen individual further has a 5 % probability of being mutated.

For the subsequent generation of the GA, the α strongest individuals from the original pool is selected and the rest are replaced by the $\mu - \alpha$ strongest individuals from the offspring pool. Thus, there are always μ individuals at the start of a new generation.

3.4.4 Termination and iteration

The algorithm is “manually” terminated after 80 generations, as it was found that there is very little probability of a better solution being found beyond this point (more on this in the *Results* section). As genetic algorithms are stochastic in their nature and thus do not always arrive at the same solution over several runs, 100 runs are performed in total. After each run, the set of active features selected by the GA’s strongest individual are saved. After all the runs have been completed, the number of occurrences of each feature across the 100 runs is counted, and the features that were selected in at least 20 % of the runs are used in the final set of optimal questions.

3.5 Selecting Optimal Questions for Current Users

The main goal of the parameter-based question evaluation (PBQE) method is to 1. find which questions are relevant to an recurrent user and 2. which of these relevant questions should be asked in a given session. This method further allows for questions that have not been posed very often (to a specific user, or to all users) to be tested and evaluated.

3.5.1 Question parameters

5 core user- and question-specific parameters are calculated in order to find an optimal question set to be posed to a recurrent user. To clarify, the parameter values are calculated for *each user* and *each question* so that every user has an associated database of parameter values for all questions in the app. Some basic metrics and modified statistical tools are used to calculate the parameter values, based on the answers in a number of recent sessions. When a session is completed, the parameters for the questions that were answered in that session are updated and stored in the database.

The following parameters are used to describe how the k :th question has been answered by the user. Note that for type 3 and 4 questions, the parameters for the sub-alternative with the strongest relevance is used as metrics for that question. Further note that if there are fewer than 20 answers available, the parameters are calculated on the available answers.

- Expected value (μ_k) - The mean value of the 20 most recent answers, as defined by 3.3. The parameter range is $[0, 1]$.

$$\mu_k = \frac{1}{20} \sum_{i=n}^{n-20} (a_i^k), \quad i = \{n, n-1, \dots, n-20\} \quad , \quad (3.3)$$

where n is the most recent session, a_i^k is the i :th answer to the k :th question.

- Deviation from expected (d_k) - This parameter is an indicator of the user's consistency in his or her most recent answer. It is defined as the difference between the most recent answer and the expected value, as defined by equation 3.4. The parameter range is $[0, 1]$. This parameter is used to find events in the user's answer to a question, where the most recent answer unexpectedly deviates from the mean.

$$d_k = |\mu_k - a_n^k| \quad , \quad (3.4)$$

where μ_k is defined as above, and a_n^k is the most recent answer to the k :th question.

- Instability (σ_k) - Describes how stable the answers have been over the 20 most recent sessions. It is an indicator of how much the user's answers to a question vary over time. Instability is defined as the sample standard deviation, as described by 3.5. The parameter value is defined in the interval $[0, 0.5]$.

$$\sigma = \sqrt{\frac{1}{19} \sum_{i=n}^{n-20} (a_i^k - \mu_k)^2}, \quad i = \{n, n-1, \dots, n-20\} \quad , \quad (3.5)$$

where the variables are defined as previously described.

- **Relevance (r_k)** - A modified version of the sample *Pearson's product-moment correlation* of the 20 most recent answers to the k :th question and the corresponding answers to the target question (3.6). This is an indicator of how relevant the question is to the user. The parameter value is defined in the interval $[0, 1]$.

$$r_k = \begin{cases} 0, & N_k < 2 \\ \frac{\text{cov}(X_k, Y)}{\sigma_{X,k} \sigma_Y}, & N_k \geq 2 \end{cases} \quad , \quad (3.6)$$

where X_k are the answers to the k :th question, Y are the target question answers, $\text{cov}(X_k, Y)$ is the covariance of these answers, N_k is the total number of available samples for the k :th question and σ is the sample standard deviation, as defined in equation 3.5.

- **Decay (λ_k)** - This parameter describes how many sessions have passed since the k :th question was last posed to the user. The parameter value is normalised and is defined in the interval $[0, 1]$, where 0 indicates that it was asked in the previous session, and 1 indicates that it was last asked one week ago.

$$\lambda = \begin{cases} \frac{1}{7}|n_o - n|, & |n_o - n| \leq 7 \\ 1, & |n_o - n| \geq 7 \end{cases} \quad , \quad (3.7)$$

where n_o is the most recent session overall, and n is the most recent session for the k :th question.

3.5.2 Question sets

There are six main sets to which a question may belong, as shown in figure 3.2. The sets in this figure affected by the presented algorithm are *Optimal*, *Test*, *RFGA* and *Global*. The *Mandatory* and *Ask once* sets are not affected.

- **Optimal set** - The optimal set is solely based on question relevance, as previously defined in equation 3.6. If the question has a relevance higher than

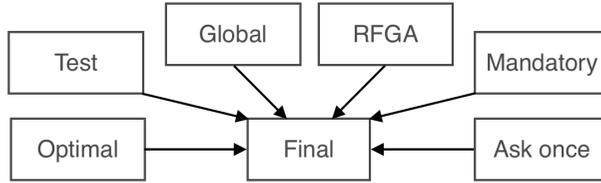


Figure 3.2: The six question sets that are used in the PBQE method.

a certain threshold ($r > 0.7$), it is included in the optimal set. The optimal set is updated after every session, so that it is always based on current data.

- *Test set* - Questions for the test set are selected from questions that are not already included in the optimal set. These remaining questions are sorted by the sum of their relevance r_k and decay λ_k , in descending order. A number of questions are selected from the top of this list, so that questions that have higher relevance, and/or has not been asked recently are prioritised. The size of the test set is affected by the total number of sessions a user has gone through, with fewer questions being tested as the user goes through more sessions (10, 5, 3 for 2-7, 8-13, ≥ 14 sessions respectively). If an event occurs, the test set is increased to the maximum size of 10 questions, regardless of how many sessions the user has gone through.

The test questions are posed to the user in two subsequent sessions, so that if the question has been asked fewer than two times overall, it is still guaranteed to be assigned values on all parameters.

- *Global set* - The global set is generated from a list sorted in ascending order based on the number of times the different questions have been posed in the app, to any user. A set of questions is chosen randomly and a subset of these questions are then selected through *tournament*, in the same fashion as selection is performed in a GA. This is done to avoid having a few questions posed repeatedly. There is a large difference in how many times questions have been posed in the app, meaning that the questions at the top of the list may have been posed several dozen times less than the questions lower on the list. If global questions would be chosen directly from the top of the list, they would be posed repeatedly over a large number of sessions until the number of answers have increased and they drop down in priority. Using tournament selection avoids this repeated selection, while still remaining biased towards selecting infrequently posed questions.

The global questions are posed to all users and the set is updated daily.

- *RFGA set* - This set consists of the questions generated by the RFGA method. This set is updated once a week.
- *Mandatory set* - This set is always posed to all users.
- *Ask once set* - This set contains demographic questions and is only posed in the first session.

3.5.3 Selecting questions to pose in an individual session

While questions that are included in the optimal set are considered relevant, they should not be posed to the user in every session (recall the requirement of avoiding posing questions too frequently). To ensure that questions are not posed too often, an additional selection method which produces the final question set for a given session is used. This selection method can be divided into four main scenarios: *first time user*, *young user*, *normal user* or *event*.

- *First time user* - A user's first session. As previously explained, questions for completely new users are selected with the RFGA method. The final question set will thus consist of questions found by the RFGA, along with a set of demographic questions which are only asked once in the app. To avoid too many questions being posed in the first session, the test set is not included.
- *Young user* - A user is defined as a young user when he or she has completed between two and seven sessions. Similar to first time users, the final set for young users will consist of questions selected by the RFGA method, with 10 test questions now also being included.
- *Normal user* - When a user has gone through more than seven sessions and no events have occurred, the questions in the optimised set will be evaluated based on their parameter values. The questions will go through a process consisting of logical steps to decide whether they are to be posed in the current session.

A number in the interval $[0,1]$ is produced by a pseudo-random number generator. The specific question will be added to the final question set if a variable, w_k is greater than the randomized number. The variable w_k is defined as

$$w_k = \begin{cases} 1, & d_k > v_k + 0.1 \\ \lambda_k, & \text{otherwise} \end{cases}, \quad (3.8)$$

The first expression in the above equation reacts to an unusually large fluctuation (defined as a deviation that is greater than the instability of the question plus a step) in the most recent answer to a question and ensures that the question is posed in the subsequent session if such a fluctuation occurs. Thus, if the user repeatedly begins to answer a question differently than before, that question will be guaranteed to be posed in subsequent sessions until the expected value has tuned in to the change and the answers no longer deviates significantly.

The second expression in the equation will reach its maximum, 1, if the question has not been posed in one week's time. If the question has been posed recently, the question is less likely to be selected. This expression ensures that questions are not posed too often.

- Event - An event is said to have occurred if the answer to the target question deviates from the expected value by a certain limit, as described by

$$event = \begin{cases} true, & d_t > |\mu_t + v_t| + 0.1 \\ false, & otherwise \end{cases}, \quad (3.9)$$

where d_t , μ_t and v_t is the deviation, expected value and instability of the target question, defined in the same way as for other questions.

If an event occurs, the final set will consist of the 20 most relevant questions from the optimal set plus an expanded test set consisting of 10 questions, regardless of how many sessions a user has gone through. In this case, the algorithm will not take any other parameters of the optimised questions into consideration, but simply pose the 20 most relevant questions. This expansion in the number of questions posed is performed to quickly identify if the reason behind the sudden change in the user's answer to the target question can be explained by the answers to other questions. Simply put, if a sudden change in the user's well-being occurs, we immediately expand our question set to gather more data and find an explanation.

It should be mentioned that as questions can be conditioned on the answer to another question, a final step is performed to retrieve all parent and child questions to the questions that were selected by the algorithm. This procedure may further increase the number of questions that are selected to be posed in a session.

To summarize, the process of selecting questions for a current user is shown graphically in figure 3.3.

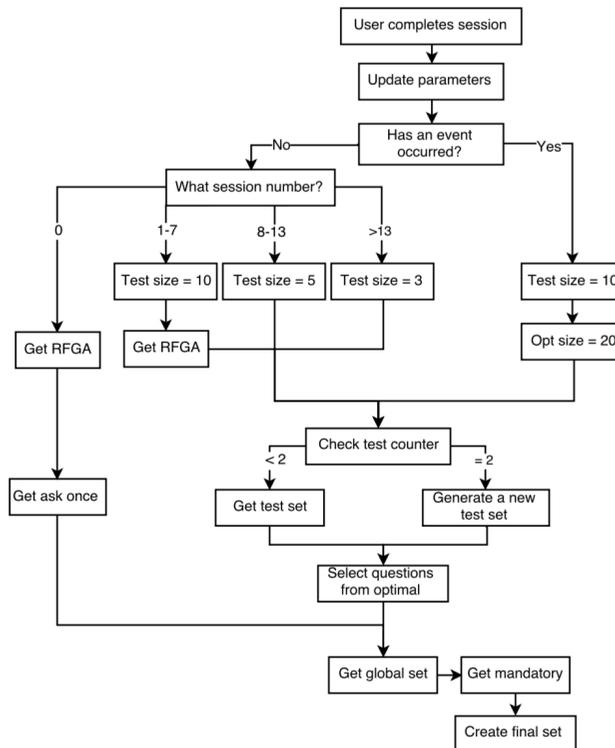


Figure 3.3: A flowchart over how questions are selected for a given session and user in A New Universe.

3.6 Evaluating the Presented Method

The presented method is evaluated in the following ways:

- Convergence of the genetic algorithm (how does fitness change over the course of evolution, how and when are optima found?)
- Dynamics of the genetic algorithm (effects of different parameter choices and fitness functions)
- Dynamics of the random forest (effects of different parameter choices and training/testing data splits)
- How well does the RFGA prediction method perform in terms of prediction error?

- Which questions are selected by the RFGA prediction method and how is this affected by data volume and quality?
- How reliable are the results of the RFGA method (how many runs of the GA needs to be performed)?
- Which and how many relevant questions are selected by the PBQE method?
- How does the size of the final question set differ between new and old users?
- How are questions chosen from the optimal set to be posed in a user's upcoming session?
- How does the PBQE method respond to events?
- How does the PBQE method affect data coverage on individual and group levels?

Chapter 4

Results

4.1 Selecting Optimal Questions for New Users

4.1.1 Dynamics of the genetic algorithm

In figure 4.1, the fitness of the RFGA prediction method over 80 generations and 100 runs can be viewed. The horizontal red line indicates the mean fitness at the start of the optimisation (0.257). The horizontal blue line indicates the mean fitness, 0.230, which is reached after 100 generations. The RMSE at this local optimum (i.e, the total fitness with the part related to the number of active features subtracted) is 0.203, which is a 19 % improvement over the course of evolution, from the initial RMSE of 0.249. It is worth noting that if the mandatory questions are included in the feature set, the mean RMSE at the end of optimisation is lowered to 0.15, as seen in table 4.1, indicating that these questions have a significant predictive power.

| Incl. Mand. Qs | | Not Incl. Mand. Qs | |
|----------------|-----------------|--------------------|-----------------|
| RMSE | Active features | RMSE | Active features |
| 0.15 | 14 | 0.23 | 14 |

Table 4.1: The RMSE of the prediction and the mean number of active features at the end of optimisation, with and without mandatory questions being included in the feature set.

The number of active features in the GA population, without fewer active features being encouraged (i.e, without the second term in equation 3.2 present) can be viewed in figure 4.2. From this figure, it can readily be seen that the

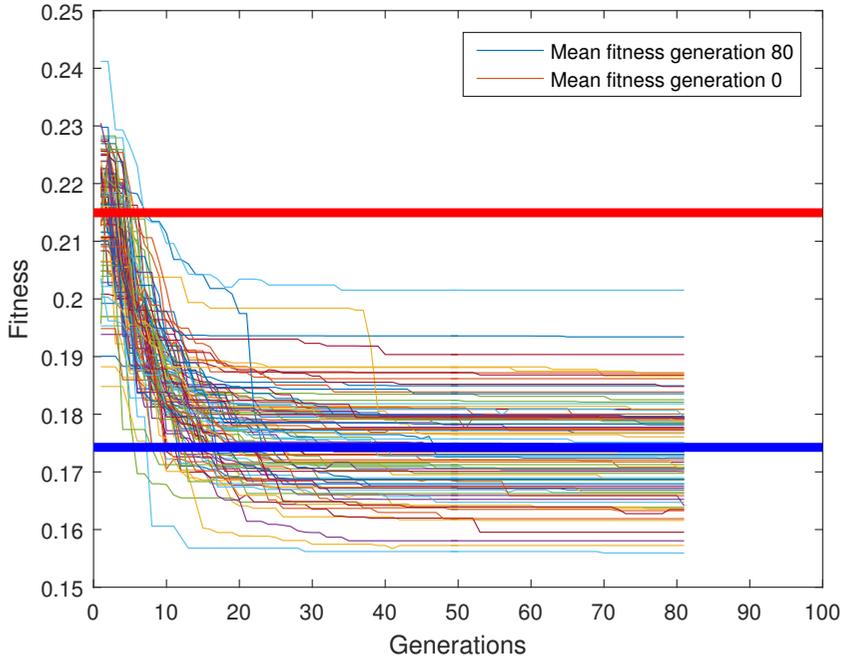


Figure 4.1: Fitness of the GA over the course of evolution and for 100 runs.

number of active features increase over the course of evolution (leveling out at around 60 - 70 active features), as a greater number of active features yield a lower RMSE.

The number of active features in individuals in the GA is influenced by the initial seeding of the individuals, as seen in figure 4.3. A greater number of active features at seeding results in a greater number of features also being active at the end of optimisation.

Finally, the execution time of the RFGA prediction is shown in table 4.2,.

| t_{ind} | n_{ind} | n_{gen} | n_{runs} | Total execution time |
|-----------|-----------|-----------|------------|----------------------|
| 0.00767 s | 100 | 80 | 100 | 6136 s |

Table 4.2: Execution time for the presented algorithm.

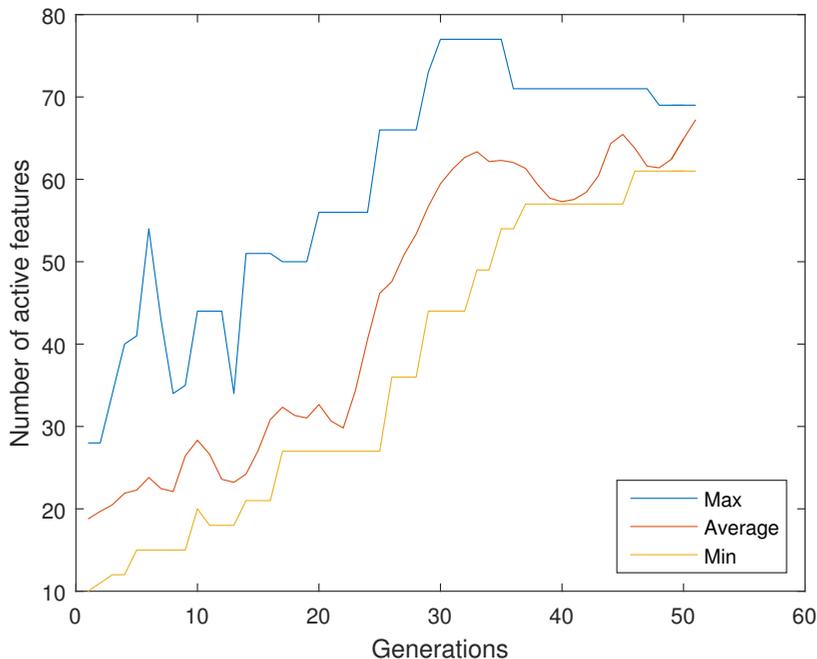


Figure 4.2: The number of active features in the population, when individuals with fewer active features are not encouraged.

4.1.2 Random forest parameters

Figure 4.4 shows the RMSE of the strongest individual at the end of optimisation as a function of different numbers of trees in the forest and different number of features to consider in the decision tree node splits.

The RMSE of predictions as a function of the percentage of data used for training can be viewed in figure 4.5,

4.1.3 Selected questions

A histogram over the number of occurrences of the selected questions can be viewed in figure 4.6. From this figure, it can be seen that of all the 125 questions that were selected in these 100 runs of the RFGA, 6 (excluding mandatory questions) fulfilled the criterion of more than 20 % occurrences. It is further worth noting that if the mandatory questions are evaluated with the RFGA prediction method,

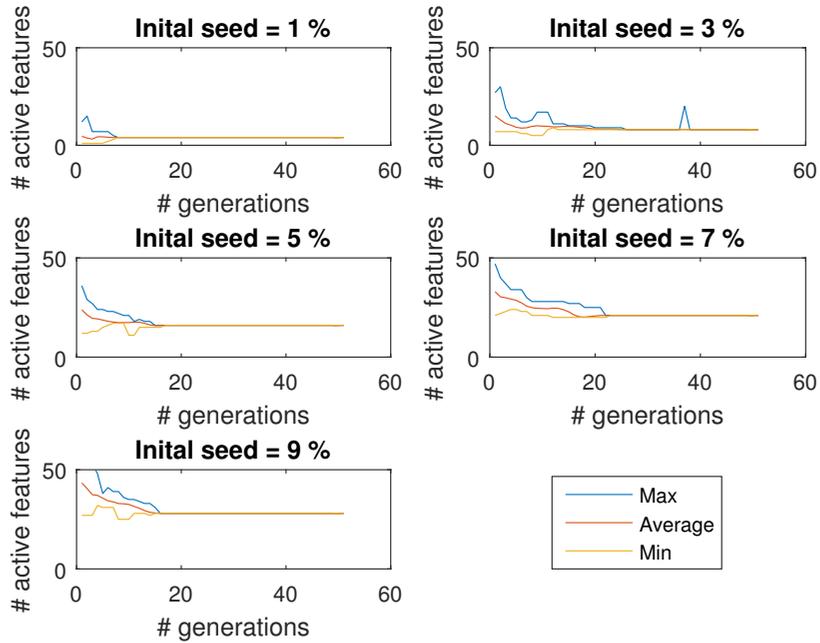


Figure 4.3: The number of active features in the population, for different initial seeds of active features.

they all occur in a majority of runs ($> 90\%$).

Some data metrics for questions that were selected in more than 20 % of the 100 RFGA runs (excluding mandatory questions) is presented in table 4.3.

| Question | Category | Type | Times selected by GA | Times posed in App |
|----------|---------------|------|----------------------|--------------------|
| 28 | Self esteem | 3 | 52 | 150 |
| 96 | Eating habits | 3 | 50 | 144 |
| 30 | Self esteem | 1 | 41 | 143 |
| 84 | General | 1 | 37 | 146 |
| 187 | Sexuality | 3 | 36 | 149 |
| 43 | Self esteem | 1 | 28 | 150 |

Table 4.3: Information on the questions that are selected by the RFGA prediction method.

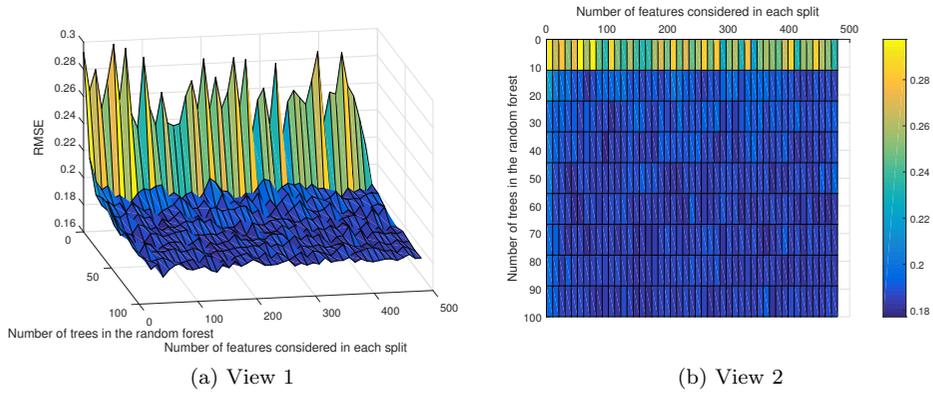


Figure 4.4: The RMSE of predictions as a function of the number of trees in the random forest and the number of features to consider in decision tree splits.

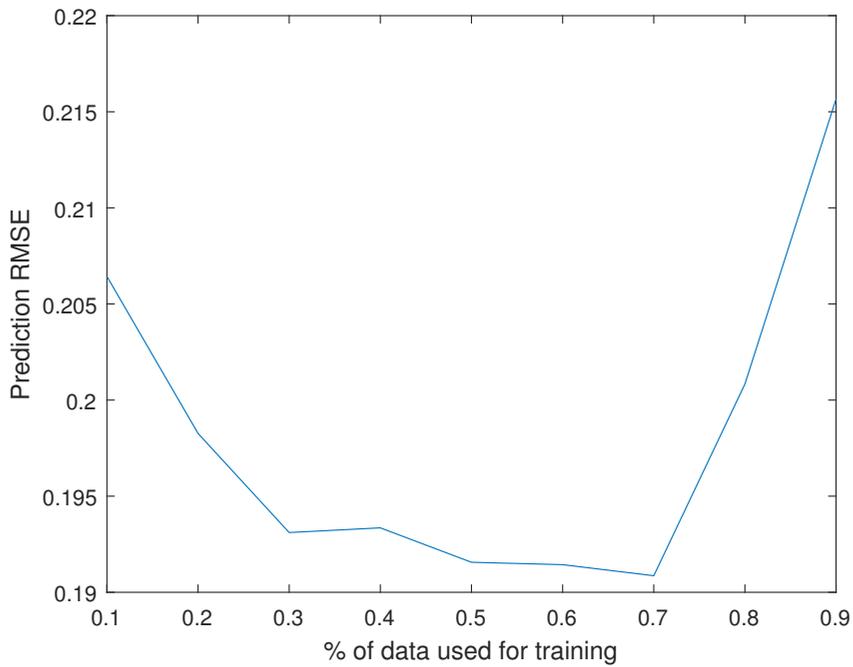


Figure 4.5: The RMSE of predictions using different percentages of the data for training. The value is the mean over 50 runs.

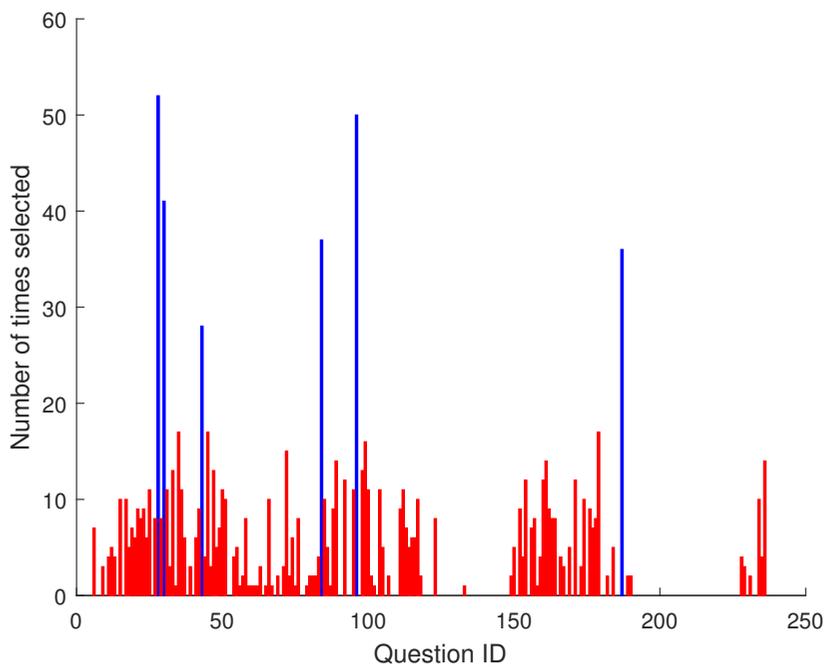


Figure 4.6: The number of times questions were selected over the 100 RFGA runs. Red indicates that the question was selected in less than the minimum 20 % of the runs, and blue that it was selected in at least 20 % of runs.

4.2 Selecting Optimal Questions for Current Users

Question statistics from users who have finished more than 13 sessions in total is included in table 4.4. It is presented here to show how equation 3.8 in section 3.5.3 affects the number of questions that are chosen from the optimal question set to be posed in the user’s next session. The table shows how many questions there are in the optimal set before and after the algorithm has been applied, as well as the portion of the original questions in optimal set that was removed.

| User ID | Sessions | Optimal, before | Optimal, after | Portion removed |
|----------|----------|-----------------|----------------|-----------------|
| 7 | 71 | 6 | 5 | 0.167 |
| 12 | 20 | 12 | 2 | 0.833 |
| 29 | 20 | 22 | 5 | 0.773 |
| 48 | 77 | 13 | 3 | 0.769 |
| 60 | 85 | 5 | 3 | 0.400 |
| 61 | 68 | 13 | 7 | 0.462 |
| 62 | 27 | 27 | 9 | 0.667 |
| 63 | 91 | 5 | 2 | 0.600 |
| 67 | 151 | 18 | 7 | 0.611 |
| 74 | 27 | 31 | 6 | 0.806 |
| 75 | 60 | 15 | 4 | 0.733 |
| 78 | 24 | 18 | 6 | 0.667 |
| 79 | 48 | 13 | 2 | 0.846 |
| average: | 60.083 | 15.417 | 4.917 | 0.624 |

Table 4.4: This table includes all users who have completed more than 13 sessions and shows the number of questions that are excluded from the current session when they are passed through equation 3.8. *Sessions* - total number of sessions, *Optimal, before* - the number of questions in the optimal set, *Optimal, after* - the number of optimal questions that will be posed to the user, *Portion removed* - the portion of questions in the optimal set removed by the algorithm.

All questions that are included in the optimal question set for a specific user are included in table 4.5. It shows which questions are selected from the optimal set, and are included in the set posed in the next session. The *relevance* and *decay* of these questions are also shown in the table.

In table 4.6 a sample selection of questions for the same user as shown in tables 4.4 and 4.5 are included. The table shows the relevance and the decay of each question, as well as which set the question is currently included in. Furthermore, the table shows in which of the three most recent sessions the question was posed,

| Question ID | Relevance | Decay | Included in final |
|-------------|-----------|-------|-------------------|
| 30 | 0.817 | 1.000 | True |
| 74 | 0.753 | 1.000 | True |
| 154 | 0.889 | 0.571 | True |
| 70 | 0.888 | 0.571 | True |
| 66 | 0.886 | 0.571 | True |
| 76 | 1.000 | 0.571 | True |
| 157 | 0.752 | 0.571 | True |
| 42 | 0.885 | 0.429 | True |
| 72 | 0.742 | 0.429 | False |
| 174 | 0.710 | 0.286 | False |
| 118 | 0.833 | 0.286 | False |
| 115 | 0.845 | 0.286 | False |
| 112 | 0.885 | 0.286 | True |
| 83 | 0.700 | 0.286 | True |
| 28 | 0.741 | 0.143 | False |
| 43 | 0.725 | 0.143 | False |
| 234 | 0.760 | 0.000 | False |
| 236 | 0.785 | 0.000 | False |
| 35 | 0.878 | 0.000 | False |

Table 4.5: All questions included in the optimal question set for a specific user. The table is sorted by *decay*. The final column, *Included in final* describes if the question will be posed in the next session.

and whether it will be posed in the next session.

| qID | Relevance | Decay | Question Set | S12 | S13 | S14 | Final |
|-----|-----------|-------|--------------|-----------|-----------|-----------|-------|
| 33 | 1.000 | 0.286 | Optimal | Asked | Not Asked | Not Asked | True |
| 34 | 0.556 | 0.429 | Test | Not Asked | Not Asked | Not Asked | True |
| 104 | 0.190 | 0.714 | Test | Not Asked | Not Asked | Not Asked | True |
| 187 | 0.572 | 0 | Global | Asked | Not Asked | Asked | True |
| 234 | 0.812 | 0 | Optimal | Asked | Asked | Asked | False |

Table 4.6: Sample questions with corresponding values for *relevance* and *decay* as well as which set the question is currently included in. The three columns S12, S13 and S14 correspond to sessions 12, 13 and 14 and shows if the question was asked in the specific session. The final column shows if the question will be posed in the next session.

Table 4.7 shows how the PBQE method handles an event. The table shows the *deviation from expected* and *instability* parameters for question 10, as well as the number of questions that are selected in the different question sets for the next session. Note that the high value of d indicates that an event has occurred and has been registered. Mandatory questions are not shown in the table but are added to the final set in a later stage.

| User ID | Sessions | d_{10} | σ_{10} | Optimal | Test | Global | Final |
|---------|----------|----------|---------------|---------|------|--------|-------|
| 96 | 15 | 0.921 | 0.248 | 20 | 10 | 5 | 23 |

Table 4.7: Shows how *deviation from expected* (d_{10}) and *instability*(σ_{10}) as well as how the number of questions in different question sets behave when an event has occurred.

Chapter 5

Discussion

5.1 Selecting Optimal Questions for New Users

5.1.1 Dynamics of the genetic algorithm

From figures 4.2 and 4.3 we can see the importance of encouraging individuals with fewer active features, and seeding appropriately. If fewer active features are not encouraged, individuals with more active features will quickly dominate the population, as the RMSE decreases when more features are available for prediction. While the prediction results seem to level out at around 60 - 70 active features, this increase in active features should be avoided as the goal is to find a small set of features that predicts sufficiently. This is effectively done by the second term of the fitness function. We can further see that the number of seeded active features greatly impacts the number of active features at the end of the optimisation. The level of seeding should thus be selected so that a reasonable number of features are activated. We can see in table 4.1 that for a seeding level of 5 %, 14 questions are selected in each run, which becomes 5-14 in the final set, after bagging and counting has been performed. Interestingly, the same number of features are active regardless of whether mandatory questions are included in the optimisation process, indicating that the mandatory questions may knock out other, less predictive questions, when they are included.

From figure 4.1, we can see that after around 60 generations, no significant improvement in fitness is achieved, as the algorithm converges at a local optimum. Mutations do have some effect after this point, but the gain in fitness is not large enough to warrant more generations being run and terminating the GA at 80 generations to save time therefore seems reasonable.

5.1.2 Random forest parameters

From figure 4.4 we can see that using 25 trees and the recommended $p/3$ considered features in each decision tree node split is a reasonable setting, as the RMSE levels out for numbers higher than these (indicating that the performance is not affected by these parameters beyond this point). From figure 4.5, we can further see that the standard 0.7 - 0.3 training/testing data split seems optimal in this prediction scenario as well.

5.1.3 Selected questions

It is apparent from several of the figures presented in the *Results* section that the performance of the RFGA method is highly dependant on two factors: 1. the predictive power of the features and 2. the number of samples available for each feature. As previously mentioned, including the mandatory questions as features in the prediction significantly decreases the RMSE. This indicates that on a group level, the answers to the mandatory questions have high predictive powers when it comes to the target question. There is also a large amount of data available on these questions, which of course improves the quality of the prediction as no data has to be imputed. However, even for less frequently asked questions it is possible to detect relevance.

From table 4.3, we can see that the questions that were selected in at least 20 % of the RFGA runs have been posed to users in A New Universe at about the same frequency (about 150 sessions or 1.3 % of all sessions). Still, the number of times the questions are selected differs. For example, question 28 was selected in 52 % of the runs, while question 43 was selected in only 28 % of runs. As the data volume for these questions is more or less equal, this indicates that question 28 is more informative than question 43. It is an interesting observation that while there is only very little data available on these questions it is still possible to detect these differences in their predictive power.

In the test runs, the RFGA selected 125 questions in total over the 100 runs. Many of these questions were selected only once or twice, indicating that their predictive power is low or that they are negatively affected by heavy imputations, but that they piggybacked on other more relevant questions. This is something which can happen in GAs, when a redundant gene is preserved from one generation to another, even though it does not improve fitness. In such scenarios, a set of features that predict very well is active in the individual, together with a few additional features that do not add anything to the prediction results (i.e, the fitness of the individual). As these weak features do not negatively impact the prediction, they will not actively reduce the fitness of the individual - they are simply redundant. Thus, they are preserved. Mutations or mating may remove them eventually, but this may take time due to the stochastic behaviour of the

GA. A good parable to redundant genes in a GA is the appendix in humans. This organ is essentially without function, but as it does not directly impact our survivability negatively, it has not been removed by evolution.

Despite the noise introduced by infrequently selected questions, some questions are clearly repeatedly selected, and are thus likely relevant. To ensure that they are indeed relevant, a large number of runs (at least 100) should be performed. This is time-consuming, and it is thus a case of a trading time for statistical power. However, as the presented algorithm is intended to be run only once a week, performing 100 runs over 100 minutes is not unreasonable.

Once again going back to table 4.3, we can see that the questions that were selected in at least 20 % of the runs have been posed around 150 times each. Based on this observation, it appears that a minimum amount of data is required for the predictive powers of questions to be detectable in the noise. As the presented algorithm will yield a bigger focus on some questions, the data volume will increase for these questions, which will in turn mean that these questions can be further validated and perhaps even yield better predictions. However, one drawback of this approach is that focusing on fewer questions will mean that we essentially create a positive feedback loop where we strengthen the position of some questions in the set as we pose them more frequently. In such a scenario, other questions which have not been posed enough times to reveal their predictive powers might never be included in the set of optimal questions. This is to some extent avoided by the global random questions and the test questions. As these questions will be posed to all users in A New Universe, the data volume available on them can quickly increase, giving them a chance to be included in the set of optimal questions (given that they are indeed predictive).

Over time, as more data is gathered on each question and the predictive powers of the questions are revealed, questions can be excluded from the app if they are deemed to be irrelevant. If one wishes to evaluate the relevance of a specific question, this question can be included in the mandatory set or in the global set and, depending on the number of active users, be fairly quickly evaluated.

One weakness of the presented algorithm, which we mentioned as a general issue with A New Universe early in this report, is that it is limited by the amount of data available. If one does not include the mandatory questions in the predictions, the smallest RMSE that can be achieved is around 0.20, whereas it is around 0.15 if the mandatory questions are included. While we cannot be certain that this increase in RMSE arise as a result of less data being available for the other questions, or if it is because the mandatory questions indeed have the highest predictive power, it is highly likely that the difference in data volume does affect the prediction. Running the RFGA several times increases the likelihood of relevant questions being correctly selected, but we still cannot evaluate the relevance of questions that have been posed too few times, as so much of the data on these

questions is imputed. In figure 4.6, we can see that some questions were selected a number of times that is just below the threshold. Some of these questions may very well have been included in the optimal set if there was more data available on them. This points to another issue with the presented algorithm, which is that the threshold for including a question in the set is arbitrarily chosen. A lower threshold essentially means that the question is less likely to be relevant while a higher threshold would ensure the relevance of the question. However, a higher threshold would negatively impact data coverage, as fewer questions are posed. Choosing a threshold is thus a question of trading data coverage for relevance. How to choose an optimal threshold is something which should be further studied.

Again, the global random questions are important here, as they have the power to quickly generate data on questions so that relevant questions can be found. Ideally, however, new questions that are added to A New Universe should be posed on a population level for some period of time to quickly gather data on them.

Another way to improve the predictions is to use a more robust method for imputing missing values. While this is outside the scope of this particular project, the imputation method is something that very likely has an impact on the results and should in the future be examined more thoroughly.

While we have now discussed issues with the prediction quality, it is worth mentioning that in this project we are not primarily interested in the quality of the prediction itself, but the separation in question relevance it provides. The best possible prediction is essentially determined by the precision and formulation of the question itself. What we have aimed to do is to compare questions and find those that are most relevant in a relative aspect, something which indeed seems possible based on the results (given that there is a minimum amount of data available).

While it is difficult to validate that the questions that are selected by the RFGA method are indeed relevant to the user under the assumption made in the method section of this report - which is that questions that can be used to predict the user's mental well-being are relevant - it does seem reasonable. For example, question 28 is "How have you been feeling in the last 24 hours, on average?". Question 43 is "How respected have you felt in the last 24 hours?". Question 30 is "How valuable have you felt in the last 24 hours?". These are rather general questions and it is intuitive that they are related to the user's well-being. Other questions are not as clear, such as question 187: "Have you viewed sexual content in the last 24 hours?". While we cannot be sure of what the connection is here, it is perhaps reasonable to expect that the user's sexual habits can have an impact on his or her mental well-being - positive and negative.

To validate the relevance of the questions, further evaluation is needed. Ideally, a survey should be conducted where the user's response to different questions is

examined. One way to do this is to include a question in each session on if the user felt that the session was meaningful. Another way is to simply conduct a survey on the test group over a longer period of time and see how their experience of A New Universe changes as the algorithm is tuned. Regardless of the method, a validation study is something that should be conducted in the future, to verify that the questions selected by the algorithm are indeed relevant.

5.2 Selecting Questions for Current Users

While questions that appear to be relevant to an individual can be detected as early as after two sessions, there are some limitations to using the absolute value of the sample Pearson moment-product correlation to detect relevance in the PBQE method. As was mentioned in the beginning of this report, one of the goals of this project was to examine the possibility of using a prediction method that does not evaluate the positive or negative connotation of the answers to individual questions, but rather explore any relevance of a question. In a question such as "How many hours did you sleep last night?", it may indeed be relevant to include this question in the optimal set, regardless of how the user has answered the question. If the user is feeling well and he or she is sleeping well, or if the user is feeling bad and he or she is not sleeping well, it may be desirable to regularly pose this question. However, for more specific questions such as "Do you feel that your alcohol habits are problematic?", it is much more difficult to remain unbiased. If the user is not feeling well, but does not have an alcohol problem, then the above question may still be detected as being relevant and posed regularly. This is, of course, problematic. There are relatively few questions of this direct type, and it may be possible to in the future implement a specific logic for these questions, but the issue of evaluating the relevance for these types of questions needs to be further refined.

For most other questions, the relevance measure appears to work well. Most questions that are included in the optimal set remain there for longer periods of time, indicating that they are correctly identified. A few questions are prematurely included and then removed, but as they are generally removed after only a few sessions, this is not a significant problem. In table 4.4, we can see that the number of questions in the optimal set generally seem to decrease as the user goes through more sessions. This again indicates that some questions are prematurely included in the optimal set, and are then removed as more data is gathered on them.

In the *Optimal, after* column in table 4.4, questions from the optimal set that are to be posed in the user's next session can be viewed. The average number of questions that are selected from the optimal set to be posed, around 5, might appear low at first glance, but these questions are not the only questions included in the final set. As previously mentioned, questions are added to the final set from

several question sets. For a user who have completed more than 13 sessions, the final question set typically consists of just under 20 questions (if $N_{test} = 3$, $N_{global} = 5$ and $N_{mand} = 5$). This number increases further as the conditions mentioned in section 2.1.2 on page 13 are applied to the questions. If we go back to the requirements in section 3.1.4 on page 41, we recall that it is not desirable to pose an excessive number of questions in a single session.

Table 4.5 shows the relationship between relevance and decay and how these parameters affect the inclusion of questions in the final set. We can see that all questions have a relevance greater than 0.7, which of course is the threshold for including a question in the optimal set. It is interesting to study the effect that the decay has on the inclusion of the question in the final set. From table 4.5, it becomes clear that a high decay is correlated with a question being included in the final set. However, we can see that questions 112 and 83 are included in the final set despite their decay being relatively low. This organic behaviour arises from the fact that the parameter-based question selection method is based on probability, and is a desired effect which introduces some variability into the questions that are posed. With this method, question sets are always relevant, but which specific questions are posed varies.

Let us study question 33 in table 4.6 more closely. We can see that this question is highly relevant and is thus included in the optimal set. It will also be included in the next session despite its low decay. A decay of 0.286 indicates that two days have passed since the question was last asked, which can be verified by looking at columns S12, S13 and S14. In contrast, question 234 is also included in the optimal set but has zero decay and is therefore not included in the final set. We can see that this question has been answered in the three most recent sessions, which indicates that it must have been included in a global set in at least one of these sessions.

Looking at questions 34 and 104, we can see that they are both included in the test set and are therefore also automatically included in the final set. They were not posed in the previous session, which tells us that they are two newly generated test questions. Since test questions should always be posed in two consecutive sessions, they will be posed in the user's following two sessions.

Finally, if we look at question 187, we can see that even though this question was posed in the most recent session (and thus has a decay of 0), it is still posed in the next session since it's included in the current global set.

From table 4.7, we can confirm that the event detection works as intended. As the answer to the target question deviates significantly from the expected value, an event is detected and the number of questions selected from the optimal set expanded, as well as the size of the test set.

5.3 Ethical Considerations

As A New Universe is primarily intended to be used by individuals suffering from some variation of mental unwellness, there are many ethical aspects to consider when developing algorithms that directly interact with these individuals. While many of the ethical challenges lies in the formulation of questions, the intended use of the application, the handling of data and the responsibility of the creators, the algorithm presented in this report also has some ethical consequences that should be addressed.

We have mentioned throughout this report that a question in A New Universe is considered relevant if the user's answers to it are in some way correlated to his or her well-being. The algorithm will focus on these questions to gather more data on aspects of the user's life that affect his or her mental health. While this is a reasonable approach in many other forms of therapy, greater care should be taken in therapeutic applications. In face-to-face therapy, the therapist has a direct interaction with the patient and can carefully construct the session so as to ensure that the patient is comfortable and relaxed. In an application, this is obviously not possible. Identifying questions that seem to be connected to the user's well-being and posing them is likely harmless - or positive - in many cases, but it may make some users uncomfortable. Some patients many not wish to be reminded of their issues, as this may make the issues feel even worse. A therapist would detect this reaction to a question and adjust the session to avoid upsetting the patient, something that is not yet possible in A New Universe. Further, some questions may be upsetting for cultural or religious reasons, such as questions concerning sexuality. While these questions may be relevant and will thus be posed with greater frequency, not all individuals may want to talk about sexuality, and may even take offense. These issues with question formulation exist regardless of the method of question selection but an increased focus on some questions may make them more apparent to the user and make him or her feel targeted.

Another factor to consider is what additional support should be available for users. If an app such as A New Universe is able to identify problem areas, how should these areas then be approached in order to help the individual? Providing feedback is problematic, and should be done with great care, so as to not give tips or recommend treatment without the involvement of a professional therapist or medical practitioner. On the other hand, recommending an app for self-reporting and then not providing any active help to the individual may also be problematic.

The lack of human interaction is a key concern in therapeutic apps, and an important factor to consider when developing novel ways of collecting data on mental health or providing support to those with mental health issues. It is difficult to define who holds responsibility for the patient and possible negative outcomes of using the app. While not directly related to the algorithm presented in this report, this is something which needs to be addressed in the future, as

more of these types of applications are developed. Great care should be taken when developing the systems, and sufficient time should be put on evaluating the effects - positive and negative - the apps may have on the users.

Chapter 6

Conclusions

In conclusion, the presented algorithm shows some promising results. All of the requirements stated at the beginning of this report have in some way been addressed, with varying - but generally positive - results.

The RFGA prediction method is able to identify a number of relevant questions on a group level, despite issues with data sparsity. Based on the formulation of these questions, it seems reasonable to assume that the selected questions are indeed relevant in predicting the users' well-being. Thus, the method may be used to validate the relevance of different questions and provide insights in how questions should be formulated in the future to provide as much information on the users as possible. It can further be used to identify an optimal set of questions to pose to new users, so as to immediately spark their interest in A New Universe.

The PBQE method, in its simplicity, is further able to personalise the interaction between A New Universe and the user by finding relevant questions while at the same time avoiding posing them too often. It is also able to detect events in the user's answers and adapt the question selection to these events. One key challenge in this method which still remains to be overcome is to find an effective way to measure question relevance without interpreting the individual answers to questions. While we have attempted to remain unbiased in this regard, we have found that it may be difficult to ensure that questions are indeed relevant without taking the positive or negative connotation of the question into consideration.

Both components in the presented algorithm do to some extent suffer from the data sparsity that is inherent to this type of data collection. For the RFGA prediction method, this means that only strong correlations are distinguishable from the noise, while weaker correlations or correlations in questions that have been answered very few times cannot be detected. In the PBQE method, this means that the measure of relevance will not be statistically reliable before a number of samples have been collected. While this method has the potential to

quickly detect true positives, it may also detect false positives that arise because of insufficient data.

Further evaluation studies should be performed to validate the effectiveness of the presented algorithm. Focus should then be put on the users' experience of the app, to validate that meaningful questions are indeed found, and that the composition of the generated question sets are optimal.

While much work remains to be done within this area, we have shown that it is indeed possible to implement machine learning algorithms to aid in the gathering of data on mental health.

Bibliography

- [1] T. Vos *et al.*, “Global, regional, and national incidence, prevalence, and years lived with disability for 301 acute and chronic diseases and injuries in 188 countries, 1990–2013: a systematic analysis for the global burden of disease study,” *The Lancet*, vol. 386, pp. 743–800, 2013.
- [2] R. Lozano *et al.*, “Global and regional mortality from 235 causes of death for 20 age groups in 1990 and 2010: a systematic analysis for the global burden of disease study,” *The Lancet*, vol. 380, pp. 2095–2128, 2012.
- [3] T. B. P. Society, “We need to talk coalition manifesto.” [Online] <http://www.bps.org.uk/psychology-public/information-public/we-need-talk-coalition/we-need-talk-coalition>, 2014. [Accessed: 2016-05-25].
- [4] B. Pettitt *et al.*, “At risk, yet dismissed: the criminal victimisation of people with mental health problems,” *London: Victim Support*, 2014.
- [5] Socialstyrelsen, “Folkhälsorapport 2009.” [Online] <http://www.socialstyrelsen.se/Lists/Artikelkatalog/Attachments/8495/2009-126-71.pdf>, 2009. [Accessed: 2016-05-26].
- [6] M. H. Foundation, “Fundamental facts about mental health.” [Online] <https://www.mentalhealth.org.uk/sites/default/files/fundamental-facts-15.pdf>, 2015. [Accessed: 2016-05-26].
- [7] C. Munsey, “At least one in 10 americans are prescribed psychotropics.” [Online] <http://www.apa.org/monitor/feb08/atleastone.aspx>, 2008. [Accessed: 2016-05-26].
- [8] S. Schneider, “3 examples of big data making a big impact on healthcare this week.” [Online] <https://blog.pivotal.io/pivotal/p-o-v/3-examples-of-big-data-making-a-big-impact-on-healthcare-this-week>, 2013. [Accessed: 2016-05-26].

- [9] R. Kessler *et al.*, “Testing a machine-learning algorithm to predict the persistence and severity of major depressive disorder from baseline self-reports,” *Molecular Psychiatry*, 2016.
- [10] Laerd, “Pearson product-moment correlation.” [Online] <https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php>. [Accessed: 2016-05-26].
- [11] A. C. Rencher and W. F. Christensen, *Methods of Multivariate Analysis*. John Wiley & Sons, 2012.
- [12] R. A. Fischer, “The use of multiple measurements in taxonomic problems,” *Annals of Eugenics*, vol. 7, pp. 179–188, 1936.
- [13] Wikipedia, “Machine learning.” [Online] https://en.wikipedia.org/wiki/Machine_learning. [Accessed: 2016-05-26].
- [14] L. Breiman *et al.*, *Classification and Regression Trees*. Wadsworth Brooks/Cole Advanced Books & Software, 1984.
- [15] M. O. Rokach, L., “Top-down induction of decision tree classifiers - a survey,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 4, pp. 476–487, 2005.
- [16] L. Breiman, “Random forests.” [Online] <https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf>. [Accessed: 2016-05-26].
- [17] C. R. Darwin, *On the Origin of Species*. John Murray, 1859.
- [18] M. Mitchell, *An Introduction to Genetic Algorithms*. MIT Press Cambridge, 1996.
- [19] R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms*. John Wiley Sons, Inc., Second ed. pp. 22-23, 2004.
- [20] MATLAB, “Peaks.” [Online] <http://se.mathworks.com/help/matlab/ref/peaks.html>. [Accessed: 2016-06-30].
- [21] R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms*. John Wiley Sons, Inc., Second ed. p. 53, 2004.
- [22] R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms*. John Wiley Sons, Inc., Second ed. pp. 36-38, 2004.
- [23] R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms*. John Wiley Sons, Inc., Second ed. pp. 38-41, 2004.

- [24] R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms*. John Wiley Sons, Inc., Second ed. pp. 41-42, 2004.
- [25] R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms*. John Wiley Sons, Inc., Second ed. pp. 57, 2004.
- [26] Wikipedia, “Singular value decomposition.” [Online] https://en.wikipedia.org/wiki/Singular_value_decomposition. [Accessed: 2016-05-30].
- [27] H. Abdi and L. J. Williams, “Principal component analysis,” vol. 4, 2010.
- [28] L. Hansen *et al.*, “Controlling feature selection in random forests of decision trees using a genetic algorithm: classification of class i mhc peptides.,” *Combinatorial Chemistry & High Throughput Screening*, vol. 12, pp. 514–519, 2009.
- [29] H. Vafaie and K. De Jong, “Genetic algorithms as a tool for feature selection in machine learning.” [Online] <https://cs.gmu.edu/~eclab/papers/TAI92.pdf>, 2009. [Accessed: 2016-05-27].
- [30] B. Bhanu and Y. Lin, “Genetic algorithm based feature selection for target detection in sar images,” *Image and Vision Computing*, vol. 21, pp. 591–608, 2003.
- [31] J. Magalhães-Mendes, “A comparative study of crossover operators for genetic algorithms to solve the job shop scheduling problem,” 2016.
- [32] J. K. Swift and J. L. Callahan, “The impact of client treatment preferences on outcome: a meta-analysis.,” *Journal of Clinical Psychology*, vol. 65, pp. 368–381, 2009.
- [33] Scholarpedia, “Evolution strategies.” [Online] http://www.scholarpedia.org/article/Evolution_strategies, 2007. [Accessed: 2016-05-27].