Utrecht University

Department of Information and Computing Science
Department of Methodology and Statistics

Thesis Project MSc Applied Data Science

A Lost Cause? Determining what variables are associated with non-response in adolescents through multiverse modelling

First examiner:
K.M. Lang

Second examiner:
M.J.L.F. Cruyff

Candidate:
L.S. Docters van Leeuwen

Student number:
6411282

August 9, 2023

Abstract

This project aimed to identify variables that consistently predict item non-response in Scottish adolescents using multiverse modeling. To implement multiverse modelling, four different state of the art algorithms were employed, along with various tuned versions. The focus of the project was on individual variable importance, although model performance was essential as well.

The results indicated that all algorithms (except for the Random Forest) performed reasonably well after tuning, depending on the target variable. The Random Forest models exhibited poor performance across all target variables and were excluded from the overall variable importance analysis. Three significant findings emerged from the study. Firstly, variables from a mental health questionnaire showed associations with missingness in multiple dependent variables. Specifically, variables related to emotions, fear, prosocial behavior, and hyperactivity were important for multiple targets. Secondly, missingness on the questionnaire itself was associated with variables related to alcohol and drug use. Lastly, missingness in the variable concerning parental supervision was strongly associated with whether the adolescent was likely tot talk to their father/carer if they had concerns.

The project had a few limitations, including some technical shortcomings. The hyperparameter optimisation process could have been more comprehensive, and the preprocessing steps were considered too harsh in retrospect. Recommendations for future research included different options to account for imbalanced data, which is ever-present in survey data. Ethical concerns about interpreting machine learning results and what research in the field on response mechanisms should emphasise were discussed as well.

Table of Contents

1 Introduction

Quantitative research within the social sciences regularly involves survey designs, where numerous questions are asked to respondents so data can be generated and analysed. While the goal generally is to have the data be as complete as possible, collected data will usually contain a fair amount of missing responses. Typically, these come in two flavours, with 'unit non-response' referring to the absence of an entire record of a respondent, and 'item non-response' indicating one or multiple items of a respondent are missing. Both present the researcher with their own unique problems, given the researchers had not intended for them to happen. In the case of unit non-response, the sample might not reflect the true population, leading to biased results among other harmful effects (Särndal & Lundström, 2005). A possible solution is weighting of the respondents that the sample did capture, as to construct a more accurate representation of the population. The other kind, item non-response, could be perceived as less abstract, as not entire records of data are missing, but only certain values of records. This essentially forces the researcher(s) to make a decision on how to treat these values, as computationally, almost no analyses can be performed. Among the social sciences, the item non-response problem deserves attention and proper treatment, but is often overlooked or neglected in reality, with crude methods like listwise deletion and mean imputation still being widely popular (Bell, Kromrey & Ferron, 2009; Savage et al., 2021). Both methods can introduce bias into the results, which could lead to inaccurate conclusions.

Fortunately, more sophisticated methods like multiple imputation by chained equations (MICE) and full information maximum likelihood (FIML) are receiving more recognition. In short, the FIML method estimates a likelihood function for each individual based on the variables that are present, thus using all data available. The MICE method works differently by using a series of regression models (where the variable with missing values is taken as the dependent variable, and the independent variables are the remaining variables) to make predictions for the missing values. A random component is added to the predictions to emulate a level of uncertainty. Normally, MICE will perform multiple cycles, constantly updating the regression models with the imputed values of the earlier cycle (Van Buuren & Groothuis-Oudshoorn, 2011). Except for a few specific situations, research has shown that MICE and FIML consistently outperform the aforementioned listwise deletion and mean imputation (Wulff & Jeppesen, 2017; Witte, Foraita & Didelez, 2022), and will produce unbiased imputations if the missing data was generated by specific mechanisms (Wulff & Jeppesen, 2017). Modern literature divides these missing data mechanisms in three distinct categories: Missing Completely At Random (MCAR), Missing At Random (MAR), and Missing Not At Random (MNAR), also called Not Data Dependent, Seen Data Dependent and Unseen Data Dependent, respectively (Van Buuren & Groothuis-Oudshoorn, 2011). In summary, when data is MCAR, the missingness is unrelated to the observed as well as the unobserved data. An example of this would be if data is not recorded because of an accidental technical difficulty during an online survey. Data is MAR when the probability of missingness is related to the observed data. Imagine a survey is held among a population to monitor anxiety, and sex of the participant is recorded as well. Assume for the sake of the example that males have a harder time answering the questions regarding anxiety, and tend to skip them. Data would then (rather counterintuitively) be called Missing At Random, as the probability of missing is dependent of sex of the respondent. Finally, MNAR implies the reason data is missing because it is related to unobserved data. In the aforementioned example about the anxiety survey, data would be MNAR if sex of the respondent had not been observed, meaning the probability of missingness on questions concerning anxiety

would now be dependent on unobserved data.

Accordingly, the sophisticated methods mentioned earlier make use of this observed data to compute imputations for the missing values. Therefore, they can only produce unbiased results if data are either MCAR or MAR. Naturally, these methods will fail to account for the non-response bias in their results if the data are MNAR, as the unobserved variables (the underlying cause of the missingness) are not present in the data. So although they are shown to perform better than listwise deletion and mean imputation (and other crude methods), researchers might not feel comfortable implementing these methods as they will have to determine what the missing data mechanism is. Especially in the social sciences, of which its students have been shown to be the least statistically literate among peers (Pan & Tang, 2004; Berndt et al., 2021), a certain hesitancy might be structurally present to consider these modern instruments. Determining what the missing data mechanism is requires the researcher to anticipate what variables might have missing values, and what other variables might account for their missingness. Data on those variables should then also be collected. Identifying which variables can account for this missingness, however, presents its own set of challenges. While the general topic of non-response is widely explored, validated information about specific constructs (like what may cause respondents to refrain from answering questions about depression) is missing, leaving the researcher to resort to speculation. To remove the need for this scientifically ambiguous process, this thesis project will aim to enrich a common knowledge base about these specific variables by predicting missingness with the assistance of algorithmic modelling. This knowledge base could lead to researchers (in particular in the social sciences) experiencing less of a barrier to implement methods like MICE and FIML. Subsequently, more accurate data and interpretations could be produced.

As the necessity and relevancy of a trustworthy scientific process is at the core of this project, principles of Open Science were applied. These principles aim to attain transparency, collaboration and accessibility by ensuring data is publicly available, research is able to be reproduced, and taking ethical considerations into account (Vicente-Saez & Martinez-Fuentes, 2018). Additionally, in consideration of possible harmful biases that may occur when the researchers degrees of freedom are not accounted for, this project utilises the novel concept of multiverse analysis (Steegen, Tuerlinckx, Gelman & Vanpaemel, 2016). The researcher degrees of freedom refer to the inherent nonuniformity of scientific experiments, as researchers can choose from a variety of different methods and approaches to conduct the data collection and analysis processes. Data dredging is one of those instances, for example, where the researcher degrees of freedom are not taken into account; Exhaustive analysis of the data and deliberately only reporting a particular set of results may create a skewed representation of the data (Smith & Ebrahim, 2002). Aiming to counteract these biases, multiverse analysis makes use of multiple methods and assesses whether results congrue among the methods used, thus increasing reliability and transparency. Regarding this project, multiverse analysis was implemented by using various (appropiate) machine learning algorithms, along with different parametrizations. Results of all of the algorithmic models were reported regardless of quality.

The data analysed in this project is from a survey concerning mental health, and alcohol and drug use among Scottish adolescents. Multiple studies have shown the majority of adults with a substance use disorder first comes in contact with alcohol and drugs as adolescents (Gutierrez & Sher, 2015), so research on these topics is crucial at this age. Moreover, adolescents have been shown to be increasingly vulnerable to mental health problems in general (Kieling et al, 2011). Accordingly, it is of significant importance to make valid inferences, which in terms of quantitative

research, requires the data to be accurate. Exploring what variables might be associated with non-response on these topics can help future research with including them, which in combination with missing data techniques decreases the bias the data might have. Hence, the research question of this thesis project is: "What variables are able to consistently predict item non-response among Scottish adolescents through multiverse modelling?"

A short description of the data is given first. This is followed by the methodology, consisting of a detailed description of the steps taken in processing the data. All of the selected algorithmic models are also elaborated on in this section. Subsequently, the results are listed, proceeded by the discussion and limitations. Ethical considerations and the conclusion complete the project. All files (except for the data, which is discussed later) and documentation are available at https://github.com/LDvLeeuwen/Thesis_LDvL.git. Note that the nature of the project was exploratory, and that reproduction is highly encouraged.

2 Data

The survey consisted of 89 questions and was conducted in 2018 by Ipsos MORI Scotland. The target population was adolescents ages 12 through 18, living in Scotland. Aside from alcohol consumption, mental health, and drug use, the survey discusses topics like parental supervision, leisure activities and friendships. To illustrate, *"Have you ever had a proper alcoholic drink - a whole drink, not just a sip?"*, *"Have you ever been offered powders or pills that are sold as legal highs?"* and *"How many close friends would you say you have?"* were among the questions included in the survey. A "Strengths and Difficulties" questionnaire was included at the end, and measured a variety of mental health constructs with statements as *"I worry a lot"* and *"I have many fears, I am easily scared"*.

The final data set provided by Ipsos consists of 635 columns, with a sample of 23.365 respondents. The reason for the large discrepancy between the number of questions and number of columns is because a lot of columns hold the same information, but are coded or grouped differently. Moreover, the presence of questions that consist of multiple statements regarding the same topic causes some questions to translate to twenty or more columns in the data. The data is publicly available (as long as the intention with the data is non-commercial; see appendix A). Skip patterns (also named 'routing') were also present in the data. Skip patterns intend to increase efficiency of a survey by only asking certain questions to a subset of the sample based on answers on earlier questions. The missing responses (if the question was indeed not asked to a respondent as a result of a skip pattern) on these items were coded as '-1' in the data. Missing values as a result of the respondent not answering the question were coded as '-9'. Naturally, only the latter are of interest in the scope of this project, as the researchers did not intend for these values to be unrecorded.

3 Methodology

In this section, an overview of the steps taken to process the data is provided, along with information about the algorithmic models and why they were deemed appropiate for the analysis. All of the processing of the data and algorithmic modelling was done in R 4.2.1. Machine learning was

implemented with the `caret` package (Kuhn et al., 2020). To ensure comprehensibility, trivial steps concerning the processing were not described, but are explained in the source code.

3.1 Data preparation

The data was prepared by removing irrelevant data, selecting which variables could be taken as dependent variables based on a cluster analysis, and finally, imputations with MICE.

3.1.1 Data cleaning

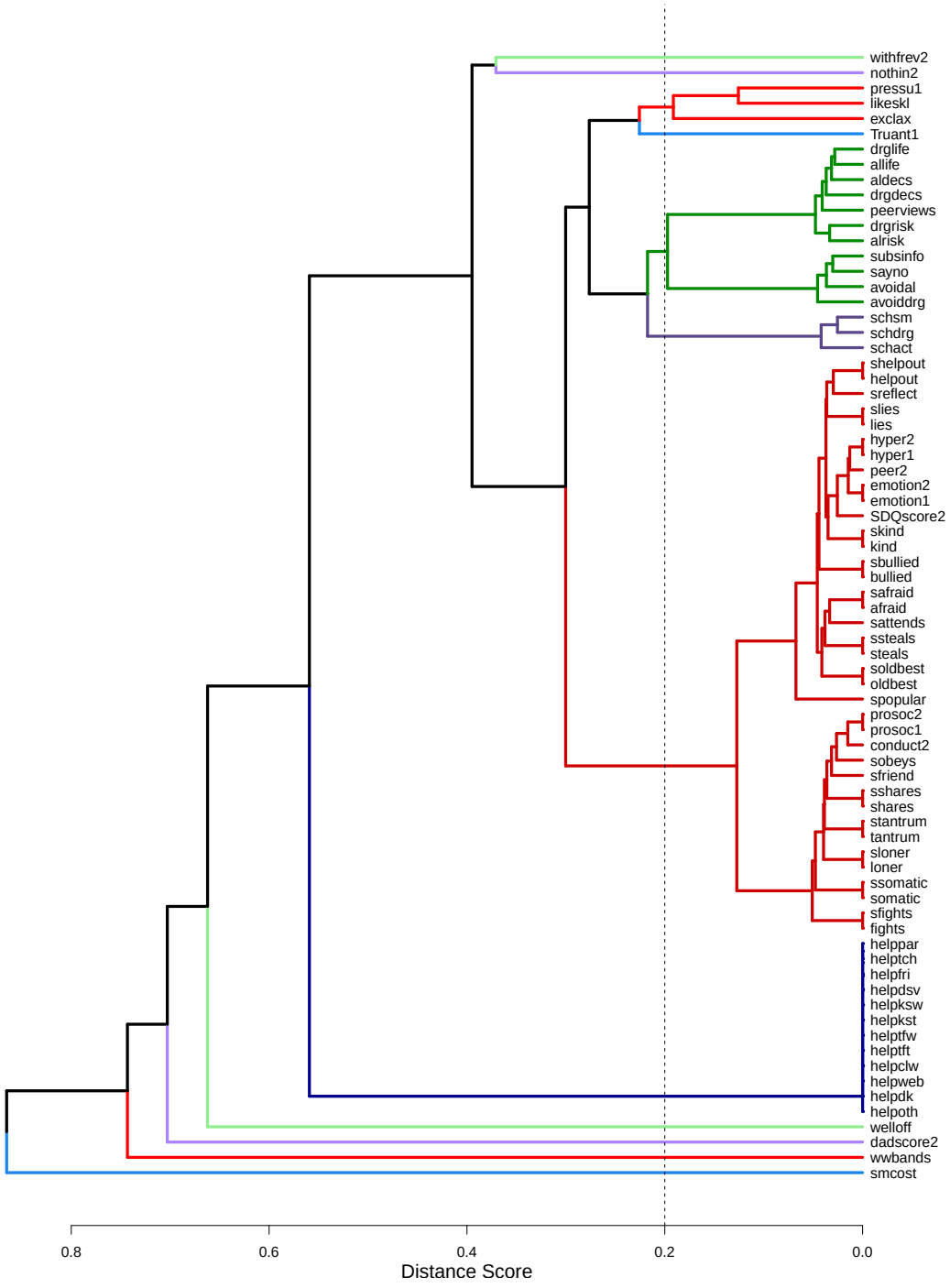
First off, data containing skip pattern questions were deemed irrelevant, and were removed accordingly. While this may appear like unnecessary data reduction, the provided codebook and example survey were examined thoroughly to support this decision. The survey was designed to include every kind of drug, for instance, which were only asked to adolescents who said that they had ever taken drugs. This results in variables that are very highly correlated, and essentially carry the same information. Another example is question 12: *"How many cigarettes did you smoke on each day in the last 7 days, ending yesterday?"*, which end up as 7 columns that all highly correlate with each other, and are therefore redundant for the analyses. Secondly, a correlation analysis was performed to discern whether variables held similar information. Correlations of each pair of variables were calculated with pairwise comparison, and as to not remove essential data, only the first variable of the pair was removed if correlations rose above a coefficient of 0.7. Lastly, a couple of variables were deleted as a result of qualitative assessment. Administrative variables like the respondent ID, variables that convey information about the same construct but were coded differently, and trivial variables about cigarette brands all fall into this category.

3.1.2 Dependent variable selection

Of the remaining 192 variables, the missingness of 74 variables were considered a possible contender as a dependent variable, as they contained more than 10% missing values. Next, a cluster analysis was performed to determine whether variables followed a similar missingness pattern. In other words, if the missing values in two or more variables occur in a corresponding order, they should not be analysed separately, since they virtually carry the same information regarding missingness.

To realise this, dummies were constructed based on the missingness of these variables. Next, a correlation matrix was calculated, essentially measuring the similarity of missingness of each pair of dummies. By converting the similarity matrix to a distance matrix, hierarchical clustering (with average linking) could be performed. The result is shown by figure 1:

Figure 1: Cluster dendrogram



A cutoff of 0.2 was applied for the distance score, leading to a total of twelve clusters. Two of those clusters were excluded from the analysis. The resulting ten clusters are listed below in table 1, along with the survey questions corresponding to the variables.

Table 1: Cluster subjects

Cluster number	Cluster subject	Variables
1	How well off the adolescent feels about the family they live with	welloff
2	Where or to whom the adolescent would want to go to if they felt that they needed help because they were using alcohol/drugs	helptch, helppar, helpfri, helpdsv, helpksw, helpkst, helptfw, helptft, helpclw, helpweb, helpdk, helpoth
3	How much information school has provided the adolescent about alcohol/drugs, according to the adolescent	alrisk, drgrisk, allife, drglife, aldecs, drgdecs, peerviews, sayno, subsinfo, avoidal, avoiddrg
4	How much advice/support school has provided the adolescent about alcohol/drugs, according to the adolescent	schdrg, schsm, schact
5	How much the adolescent is enjoying school	likeskl, pressu1, exclax
6	In the past year, how much the adolescent has skipped school	Truant1
7	Strengths & Difficulties questionnaire, various mental health constructs	somatic, shares, tantrum, loner, fights, kind, lies, bullied, helpout, steals, oldbest, afraid, ssomatic, safraid, stantrum, sobeys, sfights, slies, ssteals, sreflect, sattends, sloner, sfriend, spopular, sbullied, soldbest, sshares, skind, shelpout, emotion1, hyper1, prosoc1, emotion2, conduct2, hyper2, peer2, prosoc2, SDQscore2
8	How much the parental figures know about them and their activities, according to the adolescent	dadscore2
9	Whether the adolescent does nothing in their free time	nothin2
10	How much evenings the adolescent spends with friends in a typical week	withfrev2

Both of the excluded clusters only consisted of one dummy: One is the dummy indicating missingness in the variable `wwbands`. This variable holds a score that consists of all of the questions of the "Strengths and Difficulties" questionnaire compiled together, and is thus not based on response. The second cluster is the dummy indicating missingness in `scmost`. This variable holds information about how much adolescents thought a packet of cigarettes would cost, and was deemed irrelevant in the scope of the project.

A lot of the clusters consist of variables that were formed by the same question in the survey, which explains the similarity in missingness pattern. An example of this is cluster 3, where most of the variables represent an option to the question *"In school, how much have you learned about the following?"*, with the options being *"The risks to your health from cigarettes"*, *"The risks to your health from alcohol"*, and so forth.

One variable of the ten remaining clusters was selected arbitrarily for analysis, and are listed below in table 2, along with the amount of missingness. In preparation of the data analysis, a different data frame was formed for each cluster. These consisted of all of the preprocessed 192 variables minus the variables inside the cluster in question. Additionally, the binary dummy variable indicating the presence of missing values for the selected variable of that cluster is included in the respective data frame.

Table 2: Missingness in the dependent variables

Cluster	Variable	Missingness (in %)
1	welloff	11.26
2	helptch	11.91
3	alrisk	11.44
4	schdrg	13.46
5	likeskl	10.55
6	Truant1	13.18
7	somatic	15.53
8	dadscore2	11.92
9	nothin2	11.44
10	withfrev2	11.99

3.1.3 Imputation with MICE

MICE was used (with package `mice`) to impute the missing values in the data. Having extensively described the potential harm of applying MICE when the data might not be MCAR/MAR, this may appear tricky. Prohibiting the use of MICE, however, creates a circular argument, as the goal of this project is to uncover the underlying response mechanisms so future research can make sure potential missing data would follow a MCAR/MAR pattern. For this reason, MICE was still opted for, with additional assessment of the quality of imputations (which is mentioned later).

Since there are 10 distinct data sets for each of the dependent variables, imputations were computed for each of those frames. To make the process more efficient, the `quickpred` function was used, a method described by Stef van Buuren (2018). This method makes a selection of predictors, instead

of taking every possible variable as a predictor, which is the default. The result is a predictor matrix for the data frame, stating which variables are allowed to predict what variables through a binary system. The selection is done by calculating two types of correlations: The first correlation uses the values of the target and the predictor directly, whereas the second correlation uses the response indicator of the target and the values of the predictor. If the largest of these correlations exceeds the argument `mincor` (of which a value of 0.2 was selected), the predictor will be added to the matrix. Additionally, the procedure eliminates predictors whose proportion of usable cases fails to meet the minimum specified by the argument `minpuc`, of which a value of 0.2 was implemented. Consequently, the resulting matrix is included in the imputation process through the `predictorMatrix` argument in the `mice` function. This drastically speeds up the process, as generally the entire data frame, but roughly 25 variables are used as predictor (for each variable with missing values).

Furthermore, an `m` of 1 and a `maxit` of 3 were implemented. The `m` parameter denotes the amount of imputations MICE will make for each missing value. As variables are not selected on p-values or confidence intervals in this project (but rather if they harmonize among different algorithmic models), standard errors are less important. Therefore, a single imputation is appropiate. The `maxit` argument indicates how much iterations the MICE algorithm should run. In order to let the imputations converge, but to be mindful of the computational cost, a `maxit` of 3 was selected.

Lastly, the imputation technique for each variable. MICE requires a specification of a univariate method for each variable to be imputed. Not every method is appropiate for each type of data: For instance, logistic regression should only be used for binary variables. If these are not specified, MICE uses predictive mean matching (`pmm`) for numeric variables, logistic regression (`logreg`) for binary variables, and polytomous logistic regression (`polyreg`) for multiclass variables as a default. These default options were deemed appropiate for the data, and since all variables were converted to the correct data type beforehand (numeric as numeric, categories as factors), no specifications were given for this argument. Afterwards, this was checked with the `method` element of the imputation objects. The `loggedEvents` element (which keeps track of all shortcomings like multicollinearity) was also inspected for each set of imputations, but none were reported.

3.2 Analyses

The goal of this project can essentially be translated to a binary classification problem, as the algorithms are trained to predict missingness in a variable by taking a respective dummy (where 0 is coded as 'missing' and 1 as 'present') as the target. As table 2 visualises clearly, all of the ten target variables roughly have the same missing rate at 12%. The two classes are decidedly very imbalanced, which might pose a problem; For example, one could predict every case as 'present', and would be right in 88% of those cases. Balanced accuracy and the Matthews correlation coefficient were therefore used as the main evaluation metrics, as they both provide a more accurate assessment of model performance on imbalanced data. Balanced accuracy does this by taking both the sensitivity (the proportion of actual positive instances that are correctly identified as positive) and the specificity (the proportion of actual negative instances that are correctly identified as negative) into account. On the other hand, the Matthews correlation coefficient is calculated with all four possible outcomes of binary classification (true positives, true negatives, false positives, and false negatives), ensuring that all measures of performance are taken into consideration. Formulas are given below.

$$\text{Balanced Accuracy} = \frac{\text{Sensitivity} + \text{Specificity}}{2}$$

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}}$$

Moreover, algorithmic models whose performance has been shown to be gravely affected by class imbalance, like Logistic Regression or Linear Discriminant Analysis, were therefore not included (Brown & Mues, 2012). Algorithms who have been shown to struggle with large amounts of data (Naive Bayes, for instance) were also rejected. Furthermore, as variable evaluation is central to this project, algorithms like Support Vector Machines (which uses high-dimensional spaces, making interpretation of individual variables harder), were also not included. Utimately, the Random Forest, Neural Networks, XGBoost Regression and Adaptive Boosting algorithms were used. In the scope of the project, the models have to perform well on unseen data, so the importance of the variables belonging to those models can be generalised beyond the data used in this project. In order to evaluate how well the models perform on unseen data, the data was split in a training and a test set (with a margin of 70% for the training set). Tuning of the models was thus only applied on the training set. To reduce the chance of overfitting, and in the spirit of multiverse analysis, different training and test sets were formed for each algorithm. Overfitting occurs when a model becomes too complex and starts to "memorize" the training data rather than learning the underlying patterns. The splitting of the training and test sets was performed before each algorithm was implemented in the code, and by changing the seed (with `set.seed`) each time the data was split, the training and test sets were different for each algorithm. Three distinct tuning methods were utilized, of which a general description is listed below. Afterwards, a detailed explanation about the implementations of each algorithmic model is given.

3.2.1 Tuning methods

3.2.1.1 Hyperparameter grid search

Hyperparameters refer to the variables that define the behaviour and performance of a machine learning algorithm. These differ with each algorithm, and tuning them allow for better model performance. For this project, a hyperparameter grid search was performed for each algorithm. This is implemented by manually setting a grid with possible values for different (with the `caret::train` function), thus forming a grid. Each possible combination of hyperparameters is then tested on the training set, of which one combination will facilitate a best-performing model. The grid search was opted for as hyperparameters as the grid can be set manually, and is computationally efficient in comparison with other hyperparameter tuning methods like genetic algorithms. Since the hyperparameters differ with each algorithm, they are elaborated on in their respective sections.

3.2.1.2 Class weights

Class weights were chosen as the first method to account for the imbalanced data. They assign higher weights to the minority class, and lower weights to the majority class, aiming

to improve performance on the minority class. Inverse class weights were calculated for each dependent variable, of which a formula is listed below. They were not utilised for the Adaptive Boosting algorithm, since weighting of the classes is inherent to the algorithm itself. While tuning of the class weights is possible and offers a more refined way to calculate these weights, this is very computationally heavy, and could not be realised because of time constraints.

$$weights = \frac{1}{N_{class}}$$

3.2.1.3 Threshold tuning

Threshold tuning is the second technique used in to account for imbalanced data. In classification, an algorithmic model will assign probabilities to each class, using a threshold to ultimately predict a class label. While the standard is a threshold of 0.5, this threshold can be optimised with tuning. In the scope of the project, thresholds ranging from 0.05 to 0.5 (with intervals of 0.05) were used on the 'missing' class. For instance, if the model makes a prediction of 0.32 for the value to be 'missing', and the current threshold tested is 0.2, the model will predict the value to be 'missing'. Naturally, if a standard threshold had been applied, this prediction would have been 'present'. Threshold tuning was performed only on the training sets, and the optimal threshold was consequently used for the predictions on the unseen test set.

The hyperparameter grid search and class weights models were both validated by k-fold cross-validation. This process cuts the data in an given amount of sections, training a model on all sections except one, and testing in it on the section that was left out. Cross-validation was opted for as it prevents the given methods from overfitting. All of the tuning methods were evaluated on balanced accuracy (for the grid search and class weights models, this was implemented with the `metric` argument in `trainControl`). The stratified variant was applied to prevent the folds from having too few data points of the 'missing' class, and the data was divided in 5 folds for each instance of cross-validation.

Generally, two or three versions (for each algorithm and for each dependent variable) were tested at the same time. Threshold tuning was then applied on the best-performing model (which was evaluated with the balanced accuracy and MCC). If another threshold than 0.5 was calculated, new predictions were made. While this does not inherently change the structure of the model (or the importance of the variables), it does inform whether a model would perform better with a different threshold.

3.2.2 Algorithmic models

3.2.2.1 Random Forest

The Random Forest is an ensemble learning algorithm that combines multiple decision trees to create a powerful predictive model. Each decision tree in the random forest is trained on a random subset of the training data (on rows as well of columns of the data), and the final prediction is obtained by aggregating the predictions of all individual trees. Regarding classification, this aggregation is done by taking the majority vote. Moreover, since each tree is trained on a different subset of the data, the algorithm is less prone to overfitting. The Random Forest has been shown

to be well-suited for classification tasks as well as having the ability to handle large data sets with high dimensionality (Speiser, Miller, Tooze & Ip, 2019), and for these reasons, the algorithm was selected for the analyses.

The `randomForest` package was used for the baseline model. Then, the `caret` package with module `ranger` was used to tune hyperparameters with a grid search. The `mtry`, `splitrule` and `min.node.size` hyperparameters were tuned. `mtry` determines the number of variables randomly selected as candidates at each split point in a decision tree. Generally, larger values of mtry can improve the model's ability to capture complex relationships, but may consequently increase the risk of overfitting. In the grid search, `mtry` was tested for values 1, 5 and 15. The `splitrule` hyperparameter determines the criterion used for making splits in a decision tree. The `caret` package supports two commonly used split rules, "Gini" and "Extra trees". The "Gini" split rule, also known as Gini impurity, measures the impurity or homogeneity of a node based on the class distribution of the training samples. It aims to minimize the probability of misclassifying a randomly chosen sample. The "Extra trees" split rule, short for extremely randomized trees, is a variant of random forests that introduces additional randomness to the split point selection process. This results in faster training times but may also increase the tree's variability. Both methods were included in the grid search. Lastly, the `min.node.size` hyperparameter specifies the minimum number of samples required to create a terminal node (leaf) in a decision tree. When growing a tree, if the number of samples in a node falls below `min.node.size`, further splitting is not allowed, and the node becomes a leaf node. Increasing results in smaller trees with more generalization and smoother decision boundaries. Conversely, decreasing allows trees to capture more specific patterns and can lead to more complex models. For `min.node.size`, values 1, 5 and 10 were tested. A standard ntree of 500 was used and was left unchanged, as the `ranger` module does not support tuning of this hyperparameter (Kuhn et al., 2020).

Three versions of the Random Forest algorithm were implemented: A base version, a parameter tuned version and one with class weights and parameter tuning. Predictions on the test set were made at the same time for all three versions. Threshold tuning on the training sets was applied on all of the parameter tuned versions, as those performed best for each independent variable. Subsequently, new predictions were made with the optimal thresholds.

3.2.2.2 Neural Networks

Neural Networks are a class of deep learning models inspired by the structure and functioning of the human brain, hence the name. They consist of interconnected nodes, called neurons, which are organized in layers. Each neuron applies a mathematical transformation to its inputs and passes the result to the next layer until the final output is generated.

In order to run the analyses, categorical variables were one-hot encoded. Implementing the base version of the Neural Networks algorithm was done with the `nnet` package. The `caret` package (module `nnet`) was used to tune the `size` and `decay` parameters with a grid search. `size` determines the number of neurons in the neural network. Increasing the size value adds more neurons to the network, which can increase its capacity to learn complex patterns and relationships in the data, with the added risk of overfitting. Decreasing therefore leads to reduced performance, but less chance of overfitting. For `size`, values 5, 10, 20 and 50 were tested. The `decay` hyperparameter controls the weight decay applied. Weight decay is a regularization technique that introduces a

penalty term to the loss function during training to prevent overfitting. Increasing this penalty thus leads to a bigger chance of underfitting (the opposite of overfitting, where the model does not capture the complexities of the data). Values 0 (essentially no penalty), 0.001, 0.01 and 0.1 were included in the grid for `decay`.

A base model, a parameter tuned version and a version with inverse class weights and parameter tuning were implemented for the Neural Networks algorithm: Predictions on the test set were made at the same time for all three versions. Threshold tuning on the training sets was applied on the models that had inverse class weights and parameter tuning. For three of those, another optimal threshold than the standard 0.5 was calculated. These were models for clusters 1, 5 and 7, and for all three, the optimal threshold on the training data was 0.45. Naturally, new predictions with these threshold were made on the test set.

3.2.2.3 XGBoost Regression

The XGBoost Regression algorithm (short for Extreme Gradient Boosting) constructs an ensemble of weak prediction models in a sequential manner. New models are trained to correct the errors made by the previous ones, gradually improving the final prediction accuracy. It has been shown to excel in efficiency and performance (Ramraj, Uzir, Sunil & Banerjee, 2016), and was therefore selected for the analyses.

In order to run the analyses, categorical variables were one-hot encoded. The `xgboost` package was used to perform the analyses for the base models. Again, the grid search was realised with `caret` package (module `xgbTree`). The XGBoost algorithm has a lot of hyperparameters: the `nrounds`, `max_depth`, `eta`, `gamma`, `colsample_bytree`, `min_child_weight` and `subsample` were all included in the grid. The `nrounds` hyperparameter determines the number of boosting rounds or iterations performed during the training process. Each boosting round adds a new decision tree to the ensemble, gradually improving the model's predictive performance. For `nrounds`, values 100, 200 and 300 were opted for. `max_depth` controls the maximum depth of an individual decision tree within the ensemble, and values 3, 6 and 9 were included. Generally, for both `nrounds` and `max_depth`, as their values increase, the chance of overfitting also increases, while performance also improves.

The `eta` hyperparameter, also known as the learning rate, determines the step size at each boosting iteration, controlling the contribution of each tree to the overall ensemble. Values 0.1, 0.3 and 0.5 were opted for. `gamma` controls the minimum loss reduction required to make a further partition on a leaf node of the tree, while `min_child_weight` defines the minimum sum of instance weights required in a leaf node. For these hyperparameters, values 0, 0.1, 0.2 and 1, 3 and 5 were used respectively. Contradicting the first two hyperparameters, as values for `eta`, `gamma` and `min_child_weight` increase, the chance of overfitting is reduced. This will negatively affect the performance, though.

Finally, `colsample_bytree` determines the fraction of variables to be randomly sampled for each tree, whereas the `subsample` hyperparameter controls the fraction of training samples to be used for each boosting iteration. Values 0.6, 0.8 and 1 were included for both of them. As a general rule, setting their values beneath 1 leads to a decrease in overfitting, and improves generalisation.

Unfortunately, a version with inverse class weights could not be implemented due to an unresolved

error. Two versions of the XGBoost algorithm were thus tested: A base version and a parameter tuned version. For both versions, predictions on the test set were made at the same time. Threshold tuning on the training sets was applied on all of the parameter tuned versions, as those performed best for each independent variable. For all 10 models, a better optimal threshold than 0.5 was calculated, and new predictions were made using these improved thresholds.

3.2.2.4 Adaptive Boosting

Adaptive Boosting, also called AdaBoost, combines multiple weak classifiers to construct a strong classifier. Assigning weights to each training sample in subsequent iterations, the misclassified samples are emphasised to improve classification accuracy. Research has demonstrated its effectiveness (Margineantu & Dietterich, 1997; Feng et al., 2020), and as it is designed to handle class imbalance, the algorithm was selected for the analyses.

The baseline models for the Adaptive Boosting algorithm were implemented with package `adabag`. Tuning of hyperparameters `mfinal`, `maxdepth` and `coeflearn` was done along a grid with the `caret` package (module `AdaBoost.M1`). `mfinal` determines the maximum number of weak classifiers to be combined in the final ensemble, whereas the `maxdepth` hyperparameter specifies the maximum depth or complexity of the weak classifiers used in the ensemble. Increasing both of them can result in a more complex model. However, this also increases the risk of overfitting, especially if the dataset is small. For `mfinal`, values 50, 100 and 150 were included, whereas the values of `maxdepth` consisted of 2, 3 and 4.

Lastly, the `coeflearn` hyperparameter specifies which learning method is used in AdaBoost. `caret` supports three methods: Freund's, Breiman's, and Zhu's. Freund's method increases the weights of misclassified samples in each iteration, effectively making the subsequent weak classifiers focus more on these difficult samples. The weights are updated using an exponential function. Breiman's method, on the other hand, modifies the weight update formula to be based on the misclassification rate rather than the exponential function used in Freund's method. The weights of the misclassified samples are increased, but the increase is proportional to the misclassification rate. To conclude, Zhu's method incorporates the concept of cost-sensitive learning.

Two versions of the Adaptive Boosting algorithm were implemented: A base version and a parameter tuned version. Predictions on the test set were made at the same time for both versions. Threshold tuning on the training sets was applied on all of the parameter tuned versions, as those performed best for each independent variable. Subsequently, new predictions were made with the optimal thresholds.

4 Results

4.1 Best-performing models

The balanced accuracy metric was chosen to evaluate which version of the model performed best. A best-performing model was selected per target, per algorithm. Results are listed below in table 3, accompanied with their specificity, sensitivity, F1-score, balanced accuracy and the MCC. Results of all versions can be viewed in appendix B. Please note that for legibility, the dummy indicator

was removed from the variables in the *Target* column. As mentioned in the Methodology section, the variables themselves were not the target variables, but the missingness of those variables.

Table 3: Best model results

Cl.	Target	Model	Version	Sens.	Spec.	F1	Bal. Acc	MCC
1	welloff	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.022	0.997	0.041	0.509	0.080
1	welloff	Neural Networks	Parameter tuned + Class weights + Threshold tuned	0.662	0.635	0.292	0.648	0.192
1	welloff	XGBoost Regression	Parameter tuned + Threshold tuned	0.403	0.883	0.346	0.643	0.253
1	welloff	Adaptive Boosting	Parameter tuned + Threshold tuned	0.610	0.697	0.305	0.653	0.205
2	helptch	Random Forest	Base	0.000	1.000	NA	0.500	0.000
2	helptch	Neural Networks	Parameter tuned + Class weights	0.644	0.685	0.324	0.664	0.223
2	helptch	XGBoost Regression	Parameter tuned + Threshold tuned	0.576	0.866	0.448	0.721	0.367
2	helptch	Adaptive Boosting	Parameter tuned + Threshold tuned	0.667	0.753	0.382	0.710	0.298
3	alrisk	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.052	0.998	0.098	0.525	0.188
3	alrisk	Neural Networks	Parameter tuned + Class weights	0.709	0.771	0.407	0.740	0.339
3	alrisk	XGBoost Regression	Parameter tuned + Threshold tuned	0.759	0.963	0.743	0.861	0.710
3	alrisk	Adaptive Boosting	Parameter tuned + Threshold tuned	0.891	0.844	0.576	0.868	0.548
4	schdrg	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.266	0.990	0.401	0.628	0.427
4	schdrg	Neural Networks	Parameter tuned + Class weights	0.742	0.809	0.500	0.776	0.426
4	schdrg	XGBoost Regression	Parameter tuned + Threshold tuned	0.782	0.945	0.731	0.863	0.688

Table 3: Best model results *(continued)*

Cl.	Target	Model	Version	Sens.	Spec.	F1	Bal. Acc	MCC
4	schdrg	Adaptive Boosting	Parameter tuned + Threshold tuned	0.871	0.774	0.524	0.822	0.474
5	likeskl	Random Forest	Base	0.000	1.000	NA	0.500	0.000
5	likeskl	Neural Networks	Parameter tuned + Class weights + Threshold tuned	0.708	0.708	0.338	0.708	0.271
5	likeskl	XGBoost Regression	Parameter tuned + Threshold tuned	0.771	0.966	0.749	0.869	0.718
5	likeskl	Adaptive Boosting	Parameter tuned + Threshold tuned	0.884	0.850	0.560	0.867	0.537
6	Truant1	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.153	0.993	0.255	0.573	0.309
6	Truant1	Neural Networks	Parameter tuned + Class weights	0.664	0.736	0.390	0.700	0.291
6	Truant1	XGBoost Regression	Parameter tuned + Threshold tuned	0.755	0.911	0.645	0.833	0.591
6	Truant1	Adaptive Boosting	Parameter tuned + Threshold tuned	0.831	0.755	0.483	0.793	0.425
7	somatic	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.290	0.965	0.392	0.627	0.352
7	somatic	Neural Networks	Parameter tuned + Class weights + Threshold tuned	0.712	0.690	0.419	0.701	0.302
7	somatic	XGBoost Regression	Parameter tuned + Threshold tuned	0.631	0.884	0.558	0.757	0.470
7	somatic	Adaptive Boosting	Parameter tuned + Threshold tuned	0.810	0.729	0.493	0.769	0.408
8	dadscore2	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.521	0.958	0.569	0.739	0.519
8	dadscore2	Neural Networks	Parameter tuned + Class weights	0.660	0.867	0.499	0.763	0.430

Table 3: Best model results *(continued)*

Cl.	Target	Model	Version	Sens.	Spec.	F1	Bal. Acc	MCC
8	dadscore2	XGBoost Regression	Parameter tuned + Threshold tuned	0.740	0.909	0.613	0.825	0.562
8	dadscore2	Adaptive Boosting	Parameter tuned + Threshold tuned	0.636	0.919	0.570	0.778	0.509
9	nothin2	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.211	0.983	0.314	0.597	0.317
9	nothin2	Neural Networks	Parameter tuned + Class weights	0.687	0.779	0.405	0.733	0.333
9	nothin2	XGBoost Regression	Parameter tuned + Threshold tuned	0.744	0.923	0.635	0.833	0.588
9	nothin2	Adaptive Boosting	Parameter tuned + Threshold tuned	0.808	0.773	0.453	0.791	0.406
10	withfrev2	Random Forest	Base	0.000	1.000	NA	0.500	0.000
10	withfrev2	Neural Networks	Parameter tuned + Class weights	0.646	0.758	0.378	0.702	0.290
10	withfrev2	XGBoost Regression	Parameter tuned + Threshold tuned	0.607	0.939	0.591	0.773	0.533
10	withfrev2	Adaptive Boosting	Parameter tuned + Threshold tuned	0.780	0.772	0.451	0.776	0.393

4.2 Variable importance

The variable importance for each best-performing model version was calculated with the `varImp` function of the `caret` package. The absolute importance coefficient was recorded as well as the relative importance. The relative importance is scored on a scale from 0 to 100, where 100 indicates the variable is the most important compared to the other variables in the model. The 5 most important variables per dependent variable were listed for comprehensibility. Full results are available in the aforementioned GitHub repository.

4.2.1 Random Forest

For the Random Forest models, the variable importance was calculated with the permutation method in `caret`. Results are listed below in tables 4 and 5.

Table 4: RF importance

Cl.	Variable	Target	Imp. Scaled	Imp. Raw
1	hpcclass	welloff	100.0000	0.0048
1	hpbclass	welloff	97.8409	0.0047
1	drinkloc	welloff	83.6689	0.0040
1	aldecs	welloff	72.8403	0.0035
1	drgdecs	welloff	70.3123	0.0034
2	drgrisk	helptch	100.0000	0.0041
2	hpcclass	helptch	90.8801	0.0037
2	aldecs	helptch	89.6040	0.0037
2	hpbclass	helptch	88.2251	0.0036
2	prosoc1	helptch	86.3743	0.0035
3	hpcclass	alrisk	100.0000	0.0191
3	hpbclass	alrisk	99.8380	0.0191
3	conduct2	alrisk	65.7974	0.0126
3	ssteals	alrisk	57.9089	0.0110
3	prosoc1	alrisk	56.4673	0.0108
4	drgdecs	schdrg	100.0000	0.0237
4	allife	schdrg	92.9151	0.0220
4	aldecs	schdrg	89.1038	0.0211
4	drglife	schdrg	88.0552	0.0209
4	drgrisk	schdrg	73.0136	0.0173
5	aldecs	likeskl	100.0000	0.0071
5	drgdecs	likeskl	98.9137	0.0070
5	prosoc1	likeskl	94.3519	0.0067
5	drglife	likeskl	93.5537	0.0066
5	allife	likeskl	91.4686	0.0065

Table 5: RF importance (cont.)

Cl.	Variable	Target	Imp. Scaled	Imp. Raw
6	aldecs	Truant1	100.0000	0.0207
6	drglife	Truant1	92.4481	0.0191
6	drgdecs	Truant1	75.7765	0.0157
6	drgrisk	Truant1	74.5867	0.0154
6	hpbclass	Truant1	70.5412	0.0146
7	drglife	somatic	100.0000	0.0226
7	drgrisk	somatic	99.0952	0.0224
7	aldecs	somatic	98.8711	0.0223
7	drgdecs	somatic	97.9989	0.0221
7	allife	somatic	81.9534	0.0185
8	talkpa	dadscore2	100.0000	0.0344
8	smkdad	dadscore2	34.4578	0.0119
8	famstat2	dadscore2	17.7902	0.0061
8	drinkloc	dadscore2	15.1372	0.0052
8	noclubs	dadscore2	11.3464	0.0039
9	aldecs	nothin2	100.0000	0.0096
9	drgdecs	nothin2	89.8904	0.0087
9	alrisk	nothin2	77.2231	0.0074
9	peerviews	nothin2	73.3547	0.0071
9	drgrisk	nothin2	72.6432	0.0070
10	prosoc1	withfrev2	100.0000	0.0134
10	drglife	withfrev2	75.8245	0.0102
10	hpbclass	withfrev2	72.6262	0.0098
10	prosoc2	withfrev2	70.5576	0.0095
10	hpcclass	withfrev2	70.0593	0.0094

4.2.2 Neural Networks

For the Neural Network models, `caret` calculates variable importance based on a method by Gevrey et al (2003). As categorical variables had to be one-hot encoded to be compatible for the `caret` package, interpreting their importance is tricky, as they cannot simply be added up. Unfortunately, due to time constraints, a more primitive method was applied: Of all the categories, the highest importance was taken, and selected to represent the whole variable. Results are listed below in tables 6 and 7.

Table 6: NN importance

Cl.	Variable	Target	Imp. Scaled	Imp. Raw
1	drinkloc	welloff	100.0000	0.7801
1	sex	welloff	94.4777	0.7384
1	numeffects	welloff	87.9064	0.6888
1	glueok	welloff	78.6499	0.6189
1	wwconf	welloff	69.3243	0.5485
2	numeffects	helptch	100.0000	0.9357
2	drinkloc	helptch	90.6396	0.8504
2	peerviews	helptch	62.3272	0.5926
2	hpcclass	helptch	59.5394	0.5673
2	subuse2	helptch	53.4200	0.5115
3	skind	alrisk	100.0000	0.8292
3	shelpout	alrisk	87.0087	0.7242
3	sex	alrisk	83.1660	0.6932
3	kind	alrisk	80.4960	0.6716
3	prosoc1	alrisk	76.2686	0.6375
4	drgdecs	schdrg	100.0000	0.6952
4	skind	schdrg	97.9256	0.6814
4	drgrisk	schdrg	93.1948	0.6498
4	kind	schdrg	93.0555	0.6489
4	peerviews	schdrg	91.8574	0.6409
5	drgdecs	likeskl	100.0000	1.2532
5	peerviews	likeskl	75.7519	0.9542
5	allife	likeskl	74.0940	0.9338
5	kind	likeskl	73.5063	0.9265
5	stantrum	likeskl	64.8755	0.8201

Table 7: NN importance (cont.)

Cl.	Variable	Target	Imp. Scaled	Imp. Raw
6	drgdecs	Truant1	100.0000	0.8359
6	peerviews	Truant1	87.6048	0.7358
6	drglife	Truant1	84.9691	0.7146
6	drinkloc	Truant1	82.4473	0.6942
6	sshares	Truant1	78.4471	0.6619
7	drgdecs	somatic	100.0000	1.0438
7	alrisk	somatic	83.1354	0.8732
7	aldecs	somatic	83.0428	0.8723
7	drglife	somatic	82.8403	0.8702
7	peerviews	somatic	78.4500	0.8258
8	talkpa	dadscore2	100.0000	1.7486
8	drinkloc	dadscore2	46.0169	0.8182
8	smkdad	dadscore2	44.7720	0.7968
8	famstat2	dadscore2	35.9351	0.6444
8	sex	dadscore2	30.5307	0.5513
9	numeffects	nothin2	100.0000	0.7979
9	subuse2	nothin2	66.8763	0.5465
9	concert2	nothin2	66.2959	0.5421
9	sex	nothin2	65.7385	0.5379
9	talkma	nothin2	63.8742	0.5238
10	sex	withfrev2	100.0000	0.6145
10	stimulants2	withfrev2	92.5828	0.5712
10	concert2	withfrev2	91.3424	0.5639
10	exclax	withfrev2	90.3690	0.5582
10	frhouse2	withfrev2	83.2343	0.5165

4.2.3 XGBoost Regression

For the XGBoost Regression models, the `caret` package uses the Gain method to calculate variable importance. Like the Neural Networks, categorical variables had to be one-hot encoded. The same method was applied to aggregate the importance. Results are listed below in tables 8 and 9.

Table 8: XG importance

Cl.	Variable	Target	Imp. Scaled	Imp. Raw
1	emotion1	welloff	100.0000	6.3278
1	hyper1	welloff	87.3227	5.5256
1	drinkloc	welloff	71.2328	4.5075
1	prosoc1	welloff	63.5728	4.0228
1	whnlvskl	welloff	48.6688	3.0797
2	emotion1	helptch	100.0000	4.8899
2	prosoc1	helptch	85.8060	4.1958
2	hyper1	helptch	84.4213	4.1281
2	oldbest	helptch	60.2818	2.9477
2	whnlvskl	helptch	59.6249	2.9156
3	kind	alrisk	100.0000	6.4310
3	schsm	alrisk	94.7004	6.0902
3	prosoc1	alrisk	91.5196	5.8856
3	shares	alrisk	81.2118	5.2227
3	emotion1	alrisk	72.2302	4.6451
4	avoiddrg	schdrg	100.0000	5.2717
4	prosoc2	schdrg	95.7463	5.0475
4	prosoc1	schdrg	94.3147	4.9720
4	kind	schdrg	91.0334	4.7990
4	hyper1	schdrg	74.5794	3.9316
5	prosoc1	likeskl	100.0000	6.1727
5	lies	likeskl	96.5622	5.9605
5	hyper1	likeskl	68.6067	4.2349
5	emotion1	likeskl	56.4535	3.4847
5	prosoc2	likeskl	54.6596	3.3739

Table 9: XG importance (cont.)

Cl.	Variable	Target	Imp. Scaled	Imp. Raw
6	schsm	Truant1	100.0000	5.1284
6	prosoc1	Truant1	95.6368	4.9046
6	emotion1	Truant1	79.1009	4.0566
6	shelpout	Truant1	75.4946	3.8716
6	ssomatic	Truant1	75.3153	3.8624
7	drglife	somatic	100.0000	6.5682
7	allife	somatic	94.4637	6.2046
7	hpbclass	somatic	90.1272	5.9197
7	sayno	somatic	80.1540	5.2647
7	drgrisk	somatic	69.8938	4.5908
8	talkpa	dadscore2	100.0000	18.3981
8	hyper1	dadscore2	25.1342	4.6242
8	drinkloc	dadscore2	20.9098	3.8470
8	emotion1	dadscore2	20.0059	3.6807
8	famstat2	dadscore2	18.0991	3.3299
9	afraid	nothin2	100.0000	8.3406
9	hyper1	nothin2	66.2195	5.5231
9	emotion1	nothin2	63.4920	5.2956
9	drgrisk	nothin2	40.9123	3.4123
9	hpcclass	nothin2	36.7574	3.0658
10	schdrg	withfrev2	100.0000	6.5038
10	emotion1	withfrev2	71.3415	4.6399
10	emotion2	withfrev2	70.5429	4.5879
10	prosoc1	withfrev2	68.7499	4.4713
10	safraid	withfrev2	52.1908	3.3944

4.2.4 Adaptive Boosting

For the Adaptive Boosting models, the `caret` package uses the Gini method to calculate variable importance. Results are listed below in tables 10 and 11.

Table 10: AB importance

Cl.	Variable	Target	Imp. Scaled	Imp. Raw
1	smcost	welloff	100.0000	0.0518
1	hpbclass	welloff	45.6591	0.0236
1	alevr	welloff	19.8455	0.0103
1	shelpout	welloff	18.8146	0.0097
1	sports	welloff	15.9656	0.0083
2	afraid	helptch	100.0000	0.0491
2	smcost	helptch	89.4970	0.0440
2	canok	helptch	42.1529	0.0207
2	candang	helptch	26.2294	0.0129
2	hpcclass	helptch	26.2023	0.0129
3	sshares	alrisk	100.0000	0.0531
3	afraid	alrisk	84.0927	0.0447
3	lies	alrisk	65.5940	0.0348
3	kind	alrisk	59.9066	0.0318
3	steals	alrisk	49.6581	0.0264
4	sshares	schdrg	100.0000	0.0581
4	hpbclass	schdrg	64.8672	0.0377
4	stantrum	schdrg	45.9804	0.0267
4	helpout	schdrg	45.6193	0.0265
4	drgdecs	schdrg	43.2618	0.0251
5	skind	likeskl	100.0000	0.0685
5	schdrg	likeskl	84.0167	0.0575
5	shares	likeskl	78.1317	0.0535
5	sshares	likeskl	65.7782	0.0450
5	kind	likeskl	64.1871	0.0439

Table 11: AB importance (cont.)

Cl.	Variable	Target	Imp. Scaled	Imp. Raw
6	hpbclass	Truant1	100.0000	0.0444
6	somatic	Truant1	94.8018	0.0421
6	stantrum	Truant1	78.2973	0.0348
6	smcost	Truant1	58.8714	0.0262
6	sloner	Truant1	52.8585	0.0235
7	smcost	somatic	100.0000	0.0444
7	schsm	somatic	50.3118	0.0223
7	hpcclass	somatic	40.3213	0.0179
7	schdrg	somatic	38.4403	0.0171
7	drgdecs	somatic	25.9766	0.0115
8	talkpa	dadscore2	100.0000	0.3032
8	smcost	dadscore2	9.8598	0.0299
8	hpcclass	dadscore2	8.0627	0.0244
8	smkdad	dadscore2	7.2363	0.0219
8	famstat2	dadscore2	5.9603	0.0181
9	drgdecs	nothin2	100.0000	0.0447
9	ssomatic	nothin2	87.8766	0.0393
9	bullied	nothin2	69.9896	0.0313
9	smcost	nothin2	53.4021	0.0239
9	ssteals	nothin2	41.2336	0.0184
10	kind	withfrev2	100.0000	0.0556
10	shelpout	withfrev2	59.7225	0.0332
10	schsm	withfrev2	54.6027	0.0304
10	alrisk	withfrev2	51.0458	0.0284
10	smcost	withfrev2	48.8034	0.0272

4.2.5 Overall variable importance

For each dependent variable, the mean of the importance score was calculated across each algorithm, resulting in the overall most important variables. The Random Forest algorithm was excluded from this calculation, as performance was very poor on each dependent variable. The result is below in table 12 and 13.

Table 12: Overall importance

Cl.	Target	Variable	Mean Imp. (scaled)
1	welloff	drinkloc	60.16
1	welloff	smcost	56.95
1	welloff	emotion1	48.28
1	welloff	hyper1	48.10
1	welloff	prosoc1	37.61
2	helptch	smcost	65.22
2	helptch	afraid	48.55
2	helptch	drinkloc	47.39
2	helptch	emotion1	44.58
2	helptch	prosoc1	43.73
3	alrisk	kind	80.13
3	alrisk	sshares	68.14
3	alrisk	prosoc1	58.08
3	alrisk	schsm	53.69
3	alrisk	helpout	47.36
4	schdrg	sshares	68.69
4	schdrg	hpbclass	67.38
4	schdrg	kind	62.41
4	schdrg	drgdecs	61.02
4	schdrg	drglife	56.66
5	likeskl	skind	50.71
5	likeskl	kind	48.47
5	likeskl	drgdecs	47.70
5	likeskl	prosoc1	47.07
5	likeskl	shares	46.72

Table 13: Overall importance (cont.)

Cl.	Target	Variable	Mean Imp. (scaled)
6	Truant1	drgdecs	62.30
6	Truant1	schsm	51.26
6	Truant1	hpbclass	49.78
6	Truant1	whnlvskl	48.84
6	Truant1	afraid	48.58
7	somatic	drglife	67.01
7	somatic	allife	61.90
7	somatic	smcost	56.10
7	somatic	avoiddrg	52.28
7	somatic	drgrisk	51.78
8	dadscore2	talkpa	100.00
8	dadscore2	drinkloc	22.94
8	dadscore2	famstat2	20.00
8	dadscore2	smkdad	19.67
8	dadscore2	hyper1	14.52
9	nothin2	drgdecs	57.59
9	nothin2	afraid	45.40
9	nothin2	ssomatic	42.52
9	nothin2	numeffects	37.94
9	nothin2	hpcclass	36.32
10	withfrev2	kind	59.36
10	withfrev2	schdrg	54.43
10	withfrev2	sex	51.00
10	withfrev2	schsm	45.99
10	withfrev2	shelpout	43.78

Since the raw variable names are hard to interpret, definitions of all important variables are given for each cluster. Explanations are not repeated for legibility.

Cluster 1, which conveyed the missingness in the question *"How well off would you say your family/the people you live with are?"*, was associated with the variables `drinkloc`, `smcost`, `emotion1`, `hyper1`, and `prosoc1`. The variable `drinkloc` holds information about the number of locations where the adolescent drinks, `smcost` about how much the adolescent thinks a pack of cigarettes

costs, `emotion1` about the emotional symptoms score of the adolescent, `hyper1` about the hyperactivity score of the adolescent, and `prosoc1` about prosocial behaviour.

Next was cluster 2, signifying the missingness in the multiple options to the question *"If you wanted information about drugs, who/where would you go to?"*. This was associated with the variables `smcost`, `afraid`, `drinkloc`, `emotion1`, and `prosoc1`. The variable `afraid` holds information about the statement *"I have many fears, I am easily scared"*.

Then, cluster 3, which consisted of the missingness in questions about the subject *How much information school has provided the adolescent about alcohol/drugs, according to the adolescent*, was associated with the variables, `kind`, `sshares`, `prosoc1`, `schsm`, and `helpout`. The variable `kind` holds information about whether the adolescent feels like they are kind to younger children, `sshares` about whether the adolescent usually shares with others, `schsm` about whether the adolescent feels like their school has provided them with knowledge about smoking, and `helpout` about volunteering activities of the adolescent.

Cluster 4, which consisted of the missingness in questions about the subject *How much advice/support school has provided the adolescent about alcohol/drugs, according to the adolescent*, was associated with the variables `sshares`, `hpbclass`, `kind`, `drgdecs`, and `drglife`. The variable `hpbclass` holds information about whether the adolescent thinks the statement "Injecting drugs can lead to Hepatitis B" is true, `drgdecs` whether the adolescent believes the ability to make decisions can be affected by taking drugs, and `drglife` about whether they believe taking drugs has effect on other areas of their life.

Next was cluster 5, which represented the missingness in questions about the subject *How much the adolescent is enjoying school*, was associated with the variables `skind`, `kind`, `drgdecs`, `prosoc1`, and `shares`. The variable `skind` holds information about whether the adolescent believes they are kind to others.

Cluster 6, which consisted of the missingness in the question *"In the past year, how many times did you skip or skive school?"*, was associated with the variables `drgdecs`, `schsm`, `hpbclass`, `whnlvskl`, and afraid. The variable `whnlvskl` holds information about what the adolescent will likely be doing after they have finished school.

Cluster 7 followed, which described the missingness in the "Strengths & Difficulties" questionnaire, was associated with the variables `drglife`, `allife`, `smcost`, `avoiddrg`, and `drgrisk`. The variable `allife` holds information about whether they believe alcohol has effect on other areas of their life, `avoiddrg` about how confident the adolescent feels about being able to avoid drugs, and `drgrisk` about how much the adolescent has learned in school about the risks of drugs.

Next was cluster 8, which represented the missingness in questions about the subject *How much the parental figures know about them and their activies, according to the adolescent*, was associated with the variables `talkpa`, `drinkloc`, `famstat2`, `smkdad`, and `hyper1`. The variable `talkpa` holds information about the likelihood of the adolescent speaking to their father/carer if worried about something, `famstat2` about family status (if both parents are present in their life), and `smkdad` about whether their dad smokes.

Cluster 9, which consisted of the missingness in the question whether the adolescent does nothing in their free time, was associated with the variables `drgdrecs`, `afraid`, `ssomatic`, `numeffects`,

and `hpcclass`. The variable `ssomatic` holds information about whether the adolescent regularly experiences headaches, stomach-aches or sickness.

Finally, cluster 10 consisted of the missingness in question "How much evenings the adolescent spends with friends in a typical week", was associated with the variables `kind`, `schdrg`, `sex`, `schsm`, and `shelpout`. The variable `sex` is about the sex of the adolescent.

5 Discussion

The project set out to determine whether variables were able to consistently predict item non-response in Scottish adolescents through multiverse modelling. Multiverse modelling was implemented by using four different algorithmic models, along with different tuned versions for each of those models. While the performance of the models was relevant, the individual variable importance was at the core of the project.

Overall, every algorithm except for the Random Forest performed decent, depending on the target variable. The Random Forest models had a poor performance on every target variable. Because of this reason, variables of those models were not taken into account for the overall variable importance. With the results, three notable discoveries were made: First, the variables that belonged to the "Strengths and Difficulties" questionnaire were associated with missingness in several dependent variables, since the variables `emotion1`, `afraid`, `prosoc1`, `hyper1`, and `sshares` are shown to be important variables in multiple clusters. The second notable finding is regarding cluster 7, which represented the missingness on the questionnaire itself. The five most important variables were all regarding alcohol and drugs. Lastly, missingness in `dadscore2` (cluster 8), which contained information about how much the adolescent thought their parents knew of them and their activities, was strongly associated with `talkpa`. This variable pertains to the likelihood of the adolescent speaking to their father/carer if they are worried about something, and was the most important association with missingness in `dadscore` for all four models.

A minor limitation of the project is the generalisibility of the results. Since the data only contains information about Scottish adolescents, the results might not apply to other age groups, or other nationalities for that matter. Referring to these results should thus not be done without discretion. A second limitation is related to the optimisation of the hyperparameters of the algorithmic models. The grid search was opted for since this is supported by the `caret` package in R, and requires manual implementation. However, this approach is limiting, as only a restricted amount of combinations of hyperparameters is tested. Other approaches to tune hyperparameters, like the Artifical Bee Colony algorithm (ABC), or genetic algorithms, search for optimal hyperparameters in a more complex manner, and may thus be the better option (Karaboga, Gorkemli, Ozturk & Karaboga, 2014; Alibrahim & Ludwig, 2021). Additionally, the preprocessing steps could be perceived as a limitation. In hindsight, they were too harsh, which might have impeded the process. It would have been informing to run the analyses again with minimal preprocessing, and to examine whether results would have differed significantly. Although this could unfortunately not happen due to time constraints, it is certainly valuable to take into account for future research. A final limitation is the mediocre quality of the results. This have been caused by the class imbalance of the data. Notably, all of the baseline versions of the models experienced difficulty with this, as every one of them predicted all of the unseen data as 'present', essentially not being able to discern what could

detect a value to be missing. Naturally, the balanced accuracies of these models were consistently circa 0.5.

Furthermore, it should be emphasised that while the imbalance of the data may have been the cause (or one of the causes) of the mediocre results, this should not be interpreted as a warning to future endeavors. A considerable amount of public data in the social sciences, especially surveys stored in national databases (like the LISS panel or data of the UK Data Service), will typically not contain variables with more than 15% missingness. This data should not be viewed as less relevant, as overlooking this missingness could prevent insights about crucial variables from happening. A different recommendation is concerning the two tuning methods (class weights and threshold tuning) that were used in the project to account for the imbalanced data. These two are both algorithm-based, meaning they only tweak how the selected algorithms operate. Various other sampling methods, like oversampling or SMOTE, address the imbalance problem by focussing on the data instead. It would be worthwile for future research to also make use of these approaches, though implementing these could result in biased results if done without discretion (Vandewiele et al., 2021). As a final recommendation, researchers should not hesitate to opt for smaller data sets to analyse. While a large data set can allow for greater generalisibility and statistical power (Wolf, Harrington, Clark & Miller, 2013), it significantly slowed down the progress of this project, especially with the more complex algorithms (like Neural Networks and XGBoost Regression). Considering this was more of an exploratory analysis (which does not aim to establish causality), the project would have benefited from analysing a smaller amount of data. Especially when more variables might be examined as the dependent variable, it would be more efficient to analyse multiple different smaller data sets instead of one larger one. This is also more in line with the principles of multiverse analysis (Steegen et al., 2016). Moreover, a smaller data set would have also allowed for the sophisticated hyperparameter tuning practices mentioned earlier.

The closing point of discussion is on two ethical concerns. First is the possible harm of using machine learning algorithms, or rather, what amount of influence their results should be granted. If a certain variable can consistently predict missingness (in other variables) through exploratory analyses like this project, this does not automatically mean they hold any causal property. This is a common pitfall in data science, but domain knowledge is necessary to certify findings made with machine learning algorithms (Grimmer, 2015; Suresh & Guttag, 2019). The second ethical consideration expands on the first one, and is regarding the possible consequences of the common knowledge base mentioned in the introduction, and the inferences made in research on these patterns in general. As research on these non-response patterns continues, certain conclusions could be taken out of context, which might slowly bring forth stigmas. To illustrate an exaggerated example, if a specific demographic is consistently shown to exhibit non-response, this could develop into a persistent association with this demographic. This association might in turn deter future researchers from making an effort to include this demographic. Since the goal is to scientifically find evidence for these patterns, the intention of this example is not to suggest that these findings should not be reported; the key is how they are reported. Ultimately, the point that should be emphasised is the importance of inclusive data collection, instead of facilitating singular associations that can easily be taken out of context. Of course, how people (or society as a whole) will interpret these results is out of the hands of the researchers to a certain extent. Nonetheless, the obligation of scientific research to communicate findings in a responsible manner should not cease to be reiterated.

5.1 Conclusion

While the project was exploratory, it could be a significant step in determining what variables might be associated with variables regarding mental health, and alcohol and drug use among adolescents. A relevant discovery was made, given that the variables belonging to the mental health questionnaire were associated with missingness in several dependent variables. Future research could consider to include mental health constructs, and perform confirmatory analyses to determine whether they might belong to true response models of other variables.

Additionally, although its limitations were discussed earlier, machine learning is excellent for these types of projects, and necessary to enrich the common knowledge base. The practice of multiverse analysis is advised to ensure transparency and reliability of the findings. It is worth emphasising for future projects that if a variable cannot be predicted well by multiple algorithms, this could also lead to a meaningful finding. The variables that the algorithms were trained with are possibly not associated with missingness in that variable, which is relevant in uncovering the true response model as well.

In conclusion, missing data is ubiquitous, and should be addressed with the right methods in order to prevent harmful biases in the data. Biased data could lead to inaccurate conclusions, which (especially in the social sciences) could have detrimental consequences. Since knowledge on response mechanisms allows future projects to better implement these methods, research on them remains highly relevant.

6 References & Appendices

Alibrahim, H., & Ludwig, S. A. (2021). Hyperparameter optimization: Comparing genetic algorithm against grid search and bayesian optimization. In 2021 IEEE Congress on Evolutionary Computation (CEC) (1551-1559).

Bell, B. A., Kromrey, J. D., & Ferron, J. M. (2009). Missing data and complex samples: The impact of listwise deletion vs. subpopulation analysis on statistical bias and hypothesis test results when data are MCAR and MAR. In Proceedings of the Joint Statistical Meetings, Survey Research Methods Section (Vol. 26, pp. 759-4770).

Berndt, M., Schmidt, F. M., Sailer, M., Fischer, F., Fischer, M. R., & Zottmann, J. M. (2021). Investigating statistical literacy and scientific reasoning & argumentation in medical-, social sciences-, and economics students. Learning and Individual Differences, 86, 101963.

Brown, I., & Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. Expert Systems with Applications, 39(3), 3446-3453.

Feng, D. C., Liu, Z. T., Wang, X. D., Chen, Y., Chang, J. Q., Wei, D. F., & Jiang, Z. M. (2020). Machine learning-based compressive strength prediction for concrete: An adaptive boosting approach. Construction and Building Materials, 230, 117000.

Gevrey, M., Dimopoulos, I., & Lek, S. (2003). Review and comparison of methods to study the contribution of variables in artificial neural network models. Ecological modelling, 160(3), 249-264.

Grimmer, J. (2015). We are all social scientists now: How big data, machine learning, and causal inference work together. PS: Political Science & Politics, 48(1), 80-83.

Grimmer, J., Roberts, M. E., & Stewart, B. M. (2021). Machine learning for social science: An agnostic approach. Annual Review of Political Science, 24, 395-419.

Gutierrez, A., & Sher, L. (2015). Alcohol and drug use among adolescents: an educational overview. International journal of adolescent medicine and health, 27(2), 207-212.

Ipsos MORI Scotland. (2020). Scottish Schools Adolescent Lifestyle and Substance Use Survey, 2018. [data collection]. UK Data Service. SN: 8615, http://doi.org/10.5255/UKDA-SN-8615-1

Karaboga, D., Gorkemli, B., Ozturk, C., & Karaboga, N. (2014). A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artificial Intelligence Review, 42, 21-57.

Kieling, C., Baker-Henningham, H., Belfer, M., Conti, G., Ertem, I., Omigbodun, O., Rohde, L.A., Srinath, S., Ulkuer, N. & Rahman, A. (2011). Child and adolescent mental health worldwide: evidence for action. The Lancet, 378(9801), 1515-1525.

Kuhn, M., Wing, J., Weston, S., Williams, A., Keefer, C., Engelhardt, A., Cooper, T., Mayer, Z., Kenkel, B., Team, R.C. (2020). Package 'caret'. The R Journal, 223(7).

Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:1506.00019.

Liss Data. (2022). Access Data. Retrieved May 25, 2023, from https://www.lissdata.nl/

Margineantu, D. D., & Dietterich, T. G. (1997, July). Pruning adaptive boosting. In ICML (Vol. 97, pp. 211-218).

Pan, W., & Tang, M. (2004). Examining the effectiveness of innovative instructional methods on reducing statistics anxiety for graduate students in the social sciences. Journal of Instructional Psychology, 31(2).

Pepinsky, T. B. (2018). A note on listwise deletion versus multiple imputation. Political Analysis, 26(4), 480-488.

Ramraj, S., Uzir, N., Sunil, R., & Banerjee, S. (2016). Experimenting XGBoost algorithm for prediction and classification of different datasets. International Journal of Control Theory and Applications, 9(40), 651-662.

Särndal, C. E., & Lundström, S. (2005). Estimation in surveys with nonresponse. John Wiley & Sons.

Savage, C., Hübner, N., Biewen, M., Nagengast, B., & Polikoff, M. S. (2021). Social studies textbook effects: Evidence from Texas. Aera Open, 7, 2332858421992345.

Smith, G. D., & Ebrahim, S. (2002). Data dredging, bias, or confounding: They can all get you into the BMJ and the Friday papers. Bmj, 325(7378), 1437-1438.

Speiser, J. L., Miller, M. E., Tooze, J., & Ip, E. (2019). A comparison of random forest variable selection methods for classification prediction modeling. Expert systems with applications, 134, 93-101.

Steegen, S., Tuerlinckx, F., Gelman, A., & Vanpaemel, W. (2016). Increasing transparency through a multiverse analysis. Perspectives on Psychological Science, 11(5), 702-712.

Suresh, H., & Guttag, J. V. (2019). A framework for understanding unintended consequences of machine learning. arXiv preprint arXiv:1901.10002, 2(8).

Van Buuren, S. (2018). Flexible imputation of missing data. CRC press.

Van Buuren, S., & Groothuis-Oudshoorn, K. (2011). mice: Multivariate imputation by chained equations in R. Journal of statistical software, 45, 1-67.

Vandewiele, G., Dehaene, I., Kovács, G., Sterckx, L., Janssens, O., Ongenae, F., De Backere, F., De Turck, F., Roelens, K., Decruyenaere, J., Van Hoecke, S., Demeester, T. (2021). Overly optimistic prediction results on imbalanced data: a case study of flaws and benefits when applying over-sampling. Artificial Intelligence in Medicine, 111, 101987.

Vicente-Saez, R., & Martinez-Fuentes, C. (2018). Open Science now: A systematic literature review for an integrated definition. Journal of business research, 88, 428-436.

Witte, J., Foraita, R., & Didelez, V. (2022). Multiple imputation and test-wise deletion for causal discovery with incomplete cohort data. Statistics in Medicine, 41(23), 4716-4743.

Wulff, J. N., & Jeppesen, L. E. (2017). Multiple imputation by chained equations in praxis: guidelines and review. Electronic Journal of Business Research Methods, 15(1), 41-56.

Wolf, E. J., Harrington, K. M., Clark, S. L., & Miller, M. W. (2013). Sample size requirements for structural equation models: An evaluation of power, bias, and solution propriety. Educational and psychological measurement, 73(6), 913-934.

6.1 Appendix A

Appendix A contains information about the data used in the project, and where it can be retrieved.

The data was provided by the UK Data Service, of which the general site can be found with this link: https://ukdataservice.ac.uk/

To gain access to the data, an e-mail has to be sent to the UK Data Service stating your intentions with the data, and which institute you belong to. Note that if the goal of the project is commercial, the UK Data Service may decide to refrain from sharing the data. The link to the specific data set can be found here: http://doi.org/10.5255/UKDA-SN-8615-1 (This reference is also included in the reference list)

6.2 Appendix B

Appendix B contains information about the model performance of all the versions of each algorithms.

Table 14: All results

Target	Model	Version	Sens.	Spec.	F1	Bal. Acc	MCC
welloff	Random Forest	Base	0.001	1.000	0.003	0.501	0.034
helptch	Random Forest	Base	0.000	1.000	NA	0.500	0.000
alrisk	Random Forest	Base	0.000	1.000	NA	0.500	0.000
schdrg	Random Forest	Base	0.005	1.000	0.011	0.503	0.068
likeskl	Random Forest	Base	0.000	1.000	NA	0.500	0.000
Truant1	Random Forest	Base	0.000	1.000	NA	0.500	0.000
somatic	Random Forest	Base	0.036	0.998	0.069	0.517	0.151
dadscore2	Random Forest	Base	0.343	0.987	0.476	0.665	0.479
nothin2	Random Forest	Base	0.000	1.000	NA	0.500	0.000
withfrev2	Random Forest	Base	0.000	1.000	NA	0.500	0.000
welloff	Random Forest	Parameter tuned	0.001	1.000	0.003	0.501	0.034
helptch	Random Forest	Parameter tuned	0.000	1.000	NA	0.500	0.000
alrisk	Random Forest	Parameter tuned	0.007	1.000	0.015	0.504	0.074
schdrg	Random Forest	Parameter tuned	0.063	1.000	0.117	0.531	0.226
likeskl	Random Forest	Parameter tuned	0.000	1.000	NA	0.500	0.000
Truant1	Random Forest	Parameter tuned	0.007	1.000	0.013	0.503	0.075
somatic	Random Forest	Parameter tuned	0.094	0.995	0.167	0.545	0.239
dadscore2	Random Forest	Parameter tuned	0.414	0.979	0.529	0.697	0.509

Table 14: All results *(continued)*

Target	Model	Version	Sens.	Spec.	F1	Bal. Acc	MCC
nothin2	Random Forest	Parameter tuned	0.000	1.000	NA	0.500	0.000
withfrev2	Random Forest	Parameter tuned	0.000	1.000	NA	0.500	0.000
welloff	Random Forest	Parameter tuned + Class weights	0.001	1.000	0.003	0.501	0.034
helptch	Random Forest	Parameter tuned + Class weights	0.000	1.000	NA	0.500	0.000
alrisk	Random Forest	Parameter tuned + Class weights	0.000	1.000	NA	0.500	0.000
schdrg	Random Forest	Parameter tuned + Class weights	0.031	1.000	0.060	0.515	0.156
likeskl	Random Forest	Parameter tuned + Class weights	0.000	1.000	NA	0.500	0.000
Truant1	Random Forest	Parameter tuned + Class weights	0.005	1.000	0.011	0.503	0.069
somatic	Random Forest	Parameter tuned + Class weights	0.081	0.996	0.147	0.538	0.222
dadscore2	Random Forest	Parameter tuned + Class weights	0.398	0.983	0.521	0.690	0.508
nothin2	Random Forest	Parameter tuned + Class weights	0.000	1.000	NA	0.500	0.000
withfrev2	Random Forest	Parameter tuned + Class weights	0.000	1.000	NA	0.500	0.000
welloff	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.022	0.997	0.041	0.509	0.080
helptch	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.001	0.999	0.002	0.500	0.007
alrisk	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.052	0.998	0.098	0.525	0.188
schdrg	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.266	0.990	0.401	0.628	0.427
likeskl	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.000	1.000	NA	0.500	0.000

Table 14: All results *(continued)*

Target	Model	Version	Sens.	Spec.	F1	Bal. Acc	MCC
Truant1	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.153	0.993	0.255	0.573	0.309
somatic	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.290	0.965	0.392	0.627	0.352
dadscore2	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.521	0.958	0.569	0.739	0.519
nothin2	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.211	0.983	0.314	0.597	0.317
withfrev2	Random Forest	Parameter tuned + Class weights + Threshold tuned	0.000	1.000	NA	0.500	0.000
welloff	Neural Networks	Base	0.000	1.000	NA	0.500	0.000
helptch	Neural Networks	Base	0.000	1.000	NA	0.500	0.000
alrisk	Neural Networks	Base	0.268	0.924	0.290	0.596	0.207
schdrg	Neural Networks	Base	0.000	1.000	NA	0.500	0.000
likeskl	Neural Networks	Base	0.000	1.000	NA	0.500	0.000
Truant1	Neural Networks	Base	0.000	1.000	NA	0.500	0.000
somatic	Neural Networks	Base	0.000	1.000	NA	0.500	0.000
dadscore2	Neural Networks	Base	0.340	0.975	0.447	0.658	0.422
nothin2	Neural Networks	Base	0.148	0.968	0.213	0.558	0.179
withfrev2	Neural Networks	Base	0.000	1.000	NA	0.500	0.000
welloff	Neural Networks	Parameter tuned	0.177	0.937	0.212	0.557	0.137
helptch	Neural Networks	Parameter tuned	0.265	0.921	0.286	0.593	0.199

Table 14: All results *(continued)*

Target	Model	Version	Sens.	Spec.	F1	Bal. Acc	MCC
alrisk	Neural Networks	Parameter tuned	0.457	0.967	0.535	0.712	0.495
schdrg	Neural Networks	Parameter tuned	0.432	0.958	0.507	0.695	0.454
likeskl	Neural Networks	Parameter tuned	0.424	0.952	0.462	0.688	0.407
Truant1	Neural Networks	Parameter tuned	0.378	0.944	0.433	0.661	0.366
somatic	Neural Networks	Parameter tuned	0.371	0.921	0.412	0.646	0.320
dadscore2	Neural Networks	Parameter tuned	0.472	0.955	0.522	0.713	0.469
nothin2	Neural Networks	Parameter tuned	0.379	0.958	0.444	0.668	0.394
withfrev2	Neural Networks	Parameter tuned	0.332	0.949	0.389	0.641	0.328
welloff	Neural Networks	Parameter tuned + Class weights	0.591	0.698	0.297	0.644	0.193
helptch	Neural Networks	Parameter tuned + Class weights	0.644	0.685	0.324	0.664	0.223
alrisk	Neural Networks	Parameter tuned + Class weights	0.709	0.771	0.407	0.740	0.339
schdrg	Neural Networks	Parameter tuned + Class weights	0.742	0.809	0.500	0.776	0.426
likeskl	Neural Networks	Parameter tuned + Class weights	0.652	0.760	0.354	0.706	0.281
Truant1	Neural Networks	Parameter tuned + Class weights	0.664	0.736	0.390	0.700	0.291
somatic	Neural Networks	Parameter tuned + Class weights	0.647	0.746	0.428	0.697	0.307
dadscore2	Neural Networks	Parameter tuned + Class weights	0.660	0.867	0.499	0.763	0.430
nothin2	Neural Networks	Parameter tuned + Class weights	0.687	0.779	0.405	0.733	0.333
withfrev2	Neural Networks	Parameter tuned + Class weights	0.646	0.758	0.378	0.702	0.290
welloff	Neural Networks	Parameter tuned + Class weights + Threshold tuned	0.662	0.635	0.292	0.648	0.192

Table 14: All results *(continued)*

Target	Model	Version	Sens.	Spec.	F1	Bal. Acc	MCC
likeskl	Neural Networks	Parameter tuned + Class weights + Threshold tuned	0.708	0.708	0.338	0.708	0.271
somatic	Neural Networks	Parameter tuned + Class weights + Threshold tuned	0.712	0.690	0.419	0.701	0.302
welloff	XGBoost Regression	Base	0.134	0.976	0.203	0.555	0.188
helptch	XGBoost Regression	Base	0.283	0.977	0.389	0.630	0.371
alrisk	XGBoost Regression	Base	0.554	0.980	0.647	0.767	0.621
schdrg	XGBoost Regression	Base	0.586	0.978	0.679	0.782	0.648
likeskl	XGBoost Regression	Base	0.544	0.975	0.618	0.759	0.587
Truant1	XGBoost Regression	Base	0.499	0.974	0.597	0.736	0.562
somatic	XGBoost Regression	Base	0.422	0.968	0.528	0.695	0.486
dadscore2	XGBoost Regression	Base	0.462	0.971	0.550	0.716	0.514
nothin2	XGBoost Regression	Base	0.406	0.980	0.521	0.693	0.502
withfrev2	XGBoost Regression	Base	0.417	0.975	0.520	0.696	0.492
welloff	XGBoost Regression	Parameter tuned	0.221	0.968	0.299	0.594	0.264
helptch	XGBoost Regression	Parameter tuned	0.338	0.965	0.423	0.652	0.382
alrisk	XGBoost Regression	Parameter tuned	0.672	0.977	0.728	0.825	0.699
schdrg	XGBoost Regression	Parameter tuned	0.622	0.977	0.704	0.800	0.673
likeskl	XGBoost Regression	Parameter tuned	0.639	0.981	0.709	0.810	0.684
Truant1	XGBoost Regression	Parameter tuned	0.560	0.976	0.651	0.768	0.618
somatic	XGBoost Regression	Parameter tuned	0.482	0.947	0.545	0.715	0.480

Table 14: All results *(continued)*

Target	Model	Version	Sens.	Spec.	F1	Bal. Acc	MCC
dadscore2	XGBoost Regression	Parameter tuned	0.491	0.968	0.569	0.730	0.530
nothin2	XGBoost Regression	Parameter tuned	0.505	0.979	0.605	0.742	0.580
withfrev2	XGBoost Regression	Parameter tuned	0.510	0.969	0.586	0.739	0.547
welloff	XGBoost Regression	Parameter tuned + Threshold tuned	0.403	0.883	0.346	0.643	0.253
helptch	XGBoost Regression	Parameter tuned + Threshold tuned	0.576	0.866	0.448	0.721	0.367
alrisk	XGBoost Regression	Parameter tuned + Threshold tuned	0.759	0.963	0.743	0.861	0.710
schdrg	XGBoost Regression	Parameter tuned + Threshold tuned	0.782	0.945	0.731	0.863	0.688
likeskl	XGBoost Regression	Parameter tuned + Threshold tuned	0.771	0.966	0.749	0.869	0.718
Truant1	XGBoost Regression	Parameter tuned + Threshold tuned	0.755	0.911	0.645	0.833	0.591
somatic	XGBoost Regression	Parameter tuned + Threshold tuned	0.631	0.884	0.558	0.757	0.470
dadscore2	XGBoost Regression	Parameter tuned + Threshold tuned	0.740	0.909	0.613	0.825	0.562
nothin2	XGBoost Regression	Parameter tuned + Threshold tuned	0.744	0.923	0.635	0.833	0.588
withfrev2	XGBoost Regression	Parameter tuned + Threshold tuned	0.607	0.939	0.591	0.773	0.533
welloff	Adaptive Boosting	Base	0.119	0.981	0.188	0.550	0.185
helptch	Adaptive Boosting	Base	0.205	0.981	0.304	0.593	0.301
alrisk	Adaptive Boosting	Base	0.620	0.981	0.702	0.801	0.676
schdrg	Adaptive Boosting	Base	0.542	0.977	0.642	0.759	0.611
likeskl	Adaptive Boosting	Base	0.559	0.981	0.649	0.770	0.625
Truant1	Adaptive Boosting	Base	0.470	0.978	0.581	0.724	0.554
somatic	Adaptive Boosting	Base	0.400	0.971	0.513	0.685	0.476

Table 14: All results *(continued)*

Target	Model	Version	Sens.	Spec.	F1	Bal. Acc	MCC
dadscore2	Adaptive Boosting	Base	0.400	0.974	0.503	0.687	0.474
nothin2	Adaptive Boosting	Base	0.444	0.984	0.566	0.714	0.551
withfrev2	Adaptive Boosting	Base	0.436	0.976	0.541	0.706	0.513
welloff	Adaptive Boosting	Parameter tuned	0.184	0.959	0.244	0.571	0.195
helptch	Adaptive Boosting	Parameter tuned	0.263	0.967	0.349	0.615	0.313
alrisk	Adaptive Boosting	Parameter tuned	0.596	0.974	0.662	0.785	0.629
schdrg	Adaptive Boosting	Parameter tuned	0.557	0.972	0.641	0.764	0.603
likeskl	Adaptive Boosting	Parameter tuned	0.566	0.978	0.647	0.772	0.620
Truant1	Adaptive Boosting	Parameter tuned	0.514	0.966	0.591	0.740	0.547
somatic	Adaptive Boosting	Parameter tuned	0.459	0.962	0.550	0.710	0.500
dadscore2	Adaptive Boosting	Parameter tuned	0.422	0.974	0.523	0.698	0.494
nothin2	Adaptive Boosting	Parameter tuned	0.446	0.971	0.535	0.709	0.500
withfrev2	Adaptive Boosting	Parameter tuned	0.445	0.970	0.535	0.708	0.498
welloff	Adaptive Boosting	Parameter tuned + Threshold tuned	0.610	0.697	0.305	0.653	0.205
helptch	Adaptive Boosting	Parameter tuned + Threshold tuned	0.667	0.753	0.382	0.710	0.298
alrisk	Adaptive Boosting	Parameter tuned + Threshold tuned	0.891	0.844	0.576	0.868	0.548
schdrg	Adaptive Boosting	Parameter tuned + Threshold tuned	0.871	0.774	0.524	0.822	0.474
likeskl	Adaptive Boosting	Parameter tuned + Threshold tuned	0.884	0.850	0.560	0.867	0.537
Truant1	Adaptive Boosting	Parameter tuned + Threshold tuned	0.831	0.755	0.483	0.793	0.425
somatic	Adaptive Boosting	Parameter tuned + Threshold tuned	0.810	0.729	0.493	0.769	0.408

Table 14: All results *(continued)*

Target	Model	Version	Sens.	Spec.	F1	Bal. Acc	MCC
dadscore2	Adaptive Boosting	Parameter tuned + Threshold tuned	0.636	0.919	0.570	0.778	0.509
nothin2	Adaptive Boosting	Parameter tuned + Threshold tuned	0.808	0.773	0.453	0.791	0.406
withfrev2	Adaptive Boosting	Parameter tuned + Threshold tuned	0.780	0.772	0.451	0.776	0.393