Supplementary Materials section

Digital Phenotyping of Smartphone Data Successfully Predicts a Broad Range of Personality Constructs

Revision 1.0 15-03-2023

Maya Hocherman1, \*Yonathan Mizrachi2, Hila Chalutz-Ben Gal3

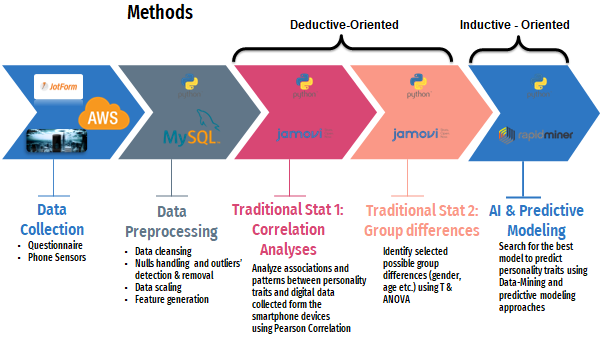
1 Tel Aviv University, Department of Industrial Engineering, Lambda Lab, Israel. hocherman@mail.tau.ac.il

2 The Max Stern Yezreel Valley Academic College / Tel Aviv University, Lambda Lab, Israel. Yoni1961@post.harvard.edu

3 Afeka Tel-Aviv Academic College of Engineering / Tel Aviv University Lambda Lab, Israel. hilab@afeka.ac.il

**Link to GIT: <https://github.com/Mayahocherman/Digital-Phenotyping-of-Smartphone-Data-Successfully-Predicts-a-Broad-Range-of-Personality-Constructs>**

# Overview: Main research workflow and technologies used for each phase.



# Stage 0: Data Collection

### ***Procedure***

Having expressed their willingness to participate in the study, participants were offered detailed participation guidelines, including links to personality questionnaires, smartphone application installation instructions and more. First, participants were asked to fill out an online questionnaire consisting of sixteen personality tests (some 500 questions) and covering some fifty-nine individual traits. The participants were asked to download and run the AWARE-light DP application (supported by our AWS back-office servers) to their personal smartphones for a week to enable collection of smartphone sensors and logs data. The data collected by the AWARE-light application consisted of Bluetooth, battery, communication, screen, and Wi-Fi networks data. Only users with Android smartphones were allowed to participate in our research. Data logs from the application were monitored by us daily to ensure continuous and uninterrupted data collection.

### ***Self-report Questionnaires***

The study included sixteen self-report questionnaires (see Table 2). The questions were assembled from sixteen well-proven, reliable, and validated previous personality research studies. Each participant’s answers were set as our” baseline” reference point for that person's personality (Kim et al., 2019). The sixteen personality constructs included in our research are described in Table 2 and are available in the GitHub repository (link above).

In addition, we controlled for age, gender, family status, education, and employment type. The participants' answers were collected and integrated with the DP smartphone data into a single database for later analyses.

### ***Behavioral Data from Smartphones***

To enable collection of the digital footprints of smartphone users, the AWARE-Light application was installed on each digital device (AWARE-Light; Digital phenotyping and experience sampling on smartphones, 2022). AWARE-Light runs in the background, so the user has no interaction with the app, which collects passive data from the user's smartphone (AWARE, 2022). Thus, we were able to collect and adjust (Kiang, 2021) the following features: general phone spec-related data, such as manufacturer and model; Bluetooth logs; battery information and monitoring of power-related events; calls and messages performed by or received by the user; screen statuses, such as turning the phone on and off, and Wi-Fi logs. Private information such as contents of calls, messages, and contact information were not collected.

REF: AWARE – Open-source Context Instrumentation Framework for Everyone. (2022). Retrieved 27 August 2022, from<https://awareframework.com/>

# Stage 1 - Data Preprocessing (done with Python - all code available in the GitHub as indicated above)

* **Data cleansing:** Twenty-eight out of 132 participants were omitted from the analyses because one or more items of the data that should have been collected were missing. For example, some participants filled in the questionnaire but had an issue downloading the app, mainly due to privacy or technical issues. Some did not finish data tracking over all seven days as was required. We also had several participants who downloaded the app but didn’t have the time to fill in the questionnaire.
* **Nulls handling:** In general, we replaced null values with the median value for the feature. For some features, we replaced null values with the mean value of the feature.
* **Outlier detection & removal:** We used the IQR (interquartile range) approach to detect outliers, and replaced them with the median for the feature.
* **Data scaling:**  All features were scaled using the Z-score method.
* **Feature generation** We went beyond the actual logs and the sensors and, by addressing the theoretical gaps, we achieved feature generation. We ourselves handled the feature generation stage, together with specialists and using related past research references. Our generated features were aggregated within two temporal aggregation levels: weekly and daily. We used 38 generated features from calls, messages, battery, screen, Wi-Fi and Bluetooth data logs. Please see our full list of conceptually generated features in this (publicly available) link: <https://bit.ly/41RipTN>. Note that in the current study we used only those features marked green. Those marked red were not used in this study. This was mostly for technical reasons and because of sampling issues (uneven collection and processing issues related to accelerometer, gyroscope, and GPS geophysical data). We plan to use these features in future research.

Please notethat all specifics of the above stages are available for review and examination in the GitHub repository.

**Where to find it on GitHub?**

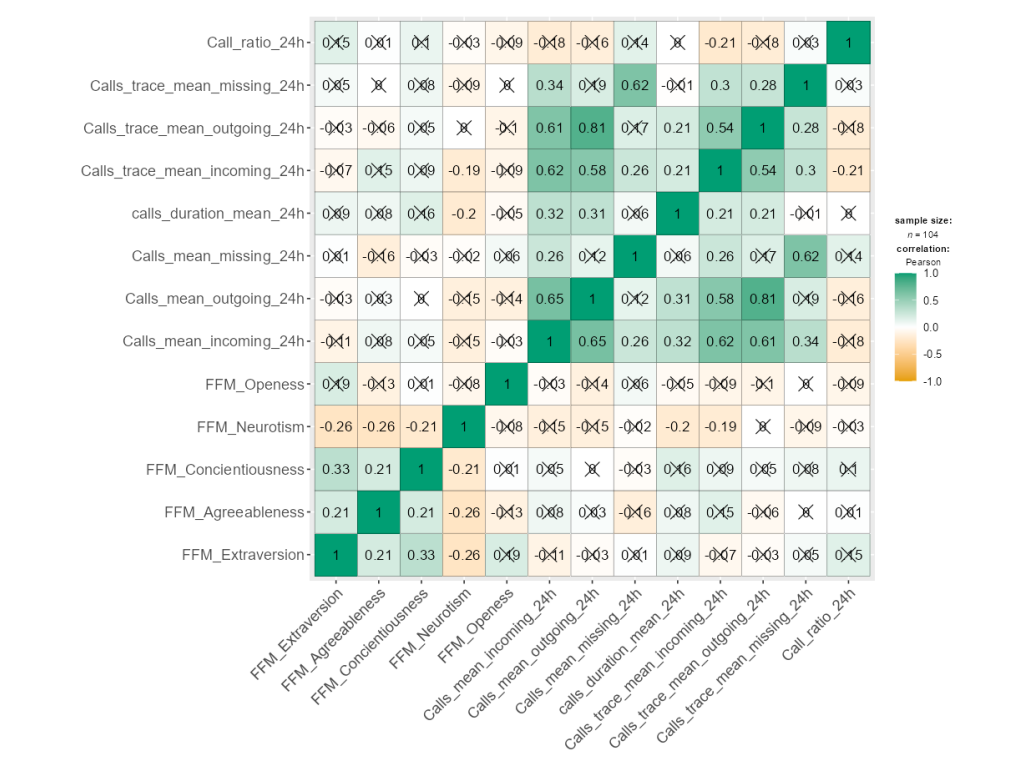
File name: Data\_Processing

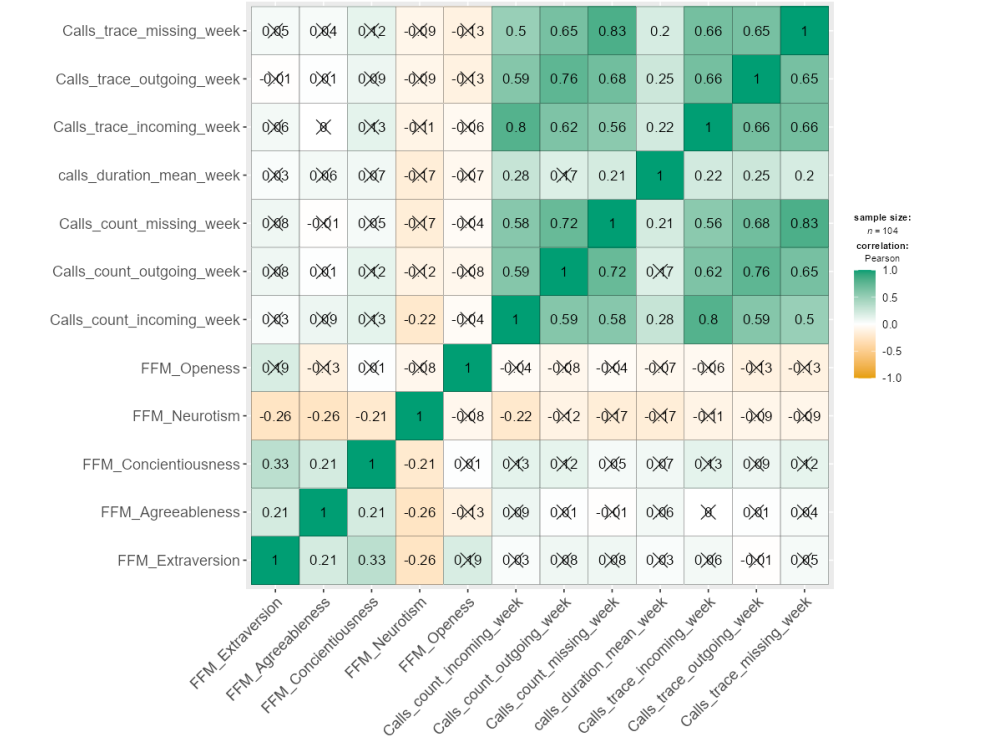
Folder: Output: Data\_processing\_output folder

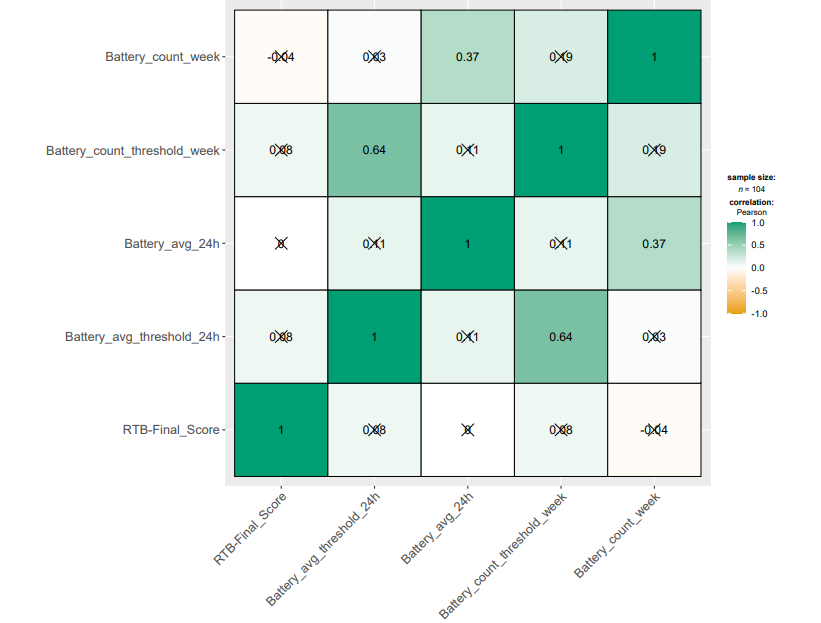
Direct link: <https://github.com/Mayahocherman/Digital-Phenotyping-of-Smartphone-Data-Successfully-Predicts-a-Broad-Range-of-Personality-Constructs/blob/main/Data_Processing.py>

# Stage 2 - Deductive analyses stage

Although not registered beforehand, and in line with the emerging [Open Science framework](https://osf.io/registries?view_only=), we did have a number of hypotheses which were tested prior to the data-mining/ML exploration phase, in order to avoid “hypotheses contamination.” We used the standard hypothesis testing approach and analyzed our data against our hypotheses using the R-based Jamovi Software. These hypotheses were deductively developed based on past literature as referenced in the Introduction section of our paper (which includes a literature review), as well as in consultation with (personality psychology) domain experts. For example, we hypothesized that the number and the length of phone calls by introverts will be significantly smaller and shorter than those of extroverts and tested for this. Similarly, we hypothesized, and subsequently tested, that individuals with phone charging behaviors that are characterized by reaching an extreme empty battery situation (beyond a certain threshold percentage of remaining battery) prior to charging their smartphone device, will demonstrate a higher average value in their Risk-Taking Behavior scores. Nevertheless, given the practical/applied nature of the current Digital Phenotyping research and the Big Data nature of our smartphone data, our focus has been on utilization of inductive approaches. All of our Jamovi data related to this stage of our research is available in the GitHub repository [here](https://github.com/Mayahocherman/Digital-Phenotyping-of-Smartphone-Data-Successfully-Predicts-a-Broad-Range-of-Personality-Constructs/tree/main/Statistical%20Analysis).





****

**Where to find it on GitHub?**

**All of our Jamovi data and processing files are available in the GitHub repository <https://github.com/Mayahocherman/Digital-Phenotyping-of-Smartphone-Data-Successfully-Predicts-a-Broad-Range-of-Personality-Constructs/tree/main/Statistical%20Analysis>**

# Stage 3 - AI & Predictive Modeling (performed with the RapidMiner non-Blackbox and tweakable Auto-model platform)

Below are the generic instruction and commands performed in this stage:

**Load Data:**

* Load the data set. Inputs are the files output from stage 1 (located in the “[Data\_processing\_output](https://github.com/Mayahocherman/Digital-Phenotyping-of-Smartphone-Data-Successfully-Predicts-a-Broad-Range-of-Personality-Constructs/tree/main/Data_processing_output)” folder of our GitHub repository). All labeled data points are delivered.

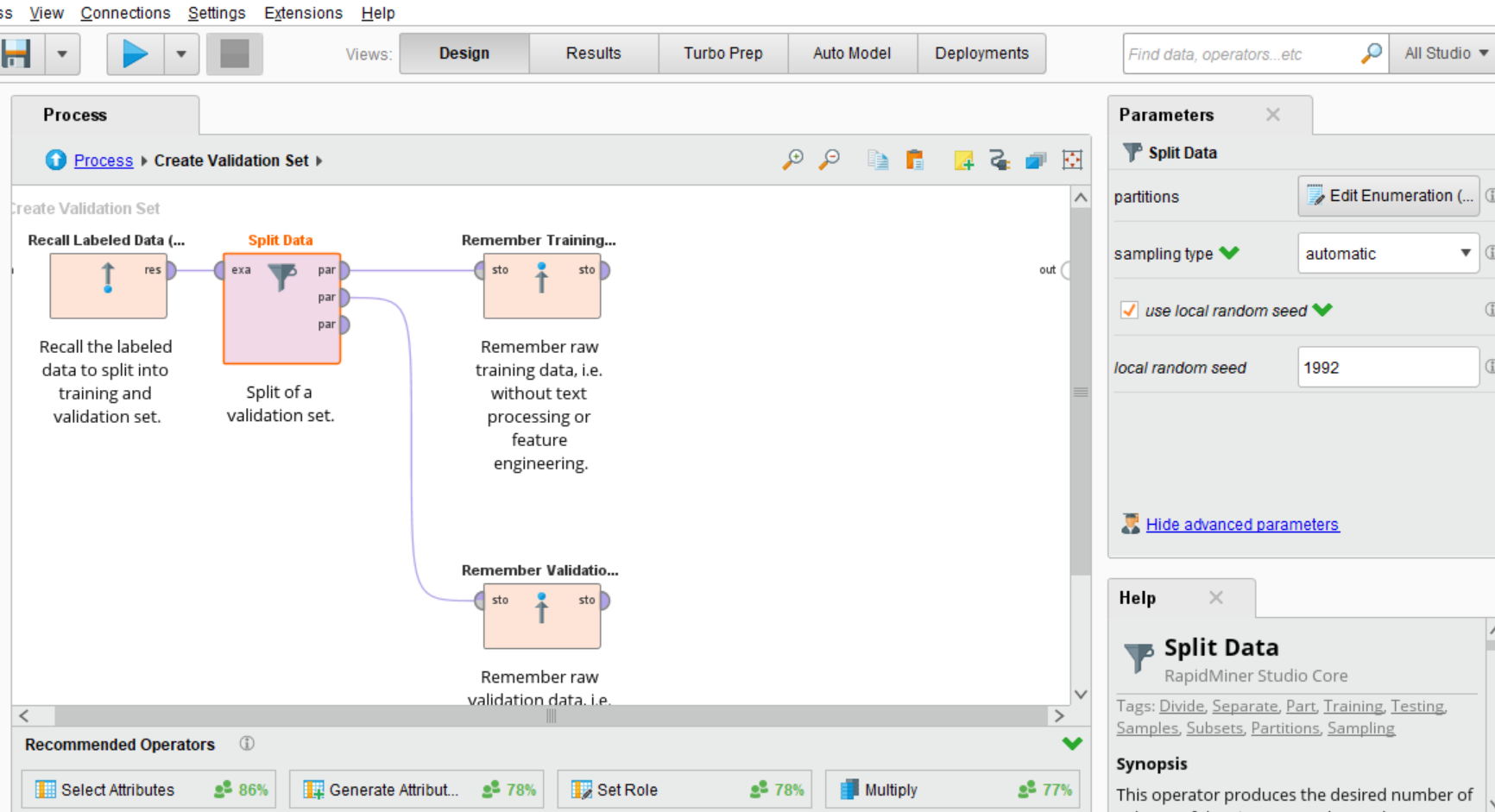
**Create Validation (training and testing models' creation) Set:**

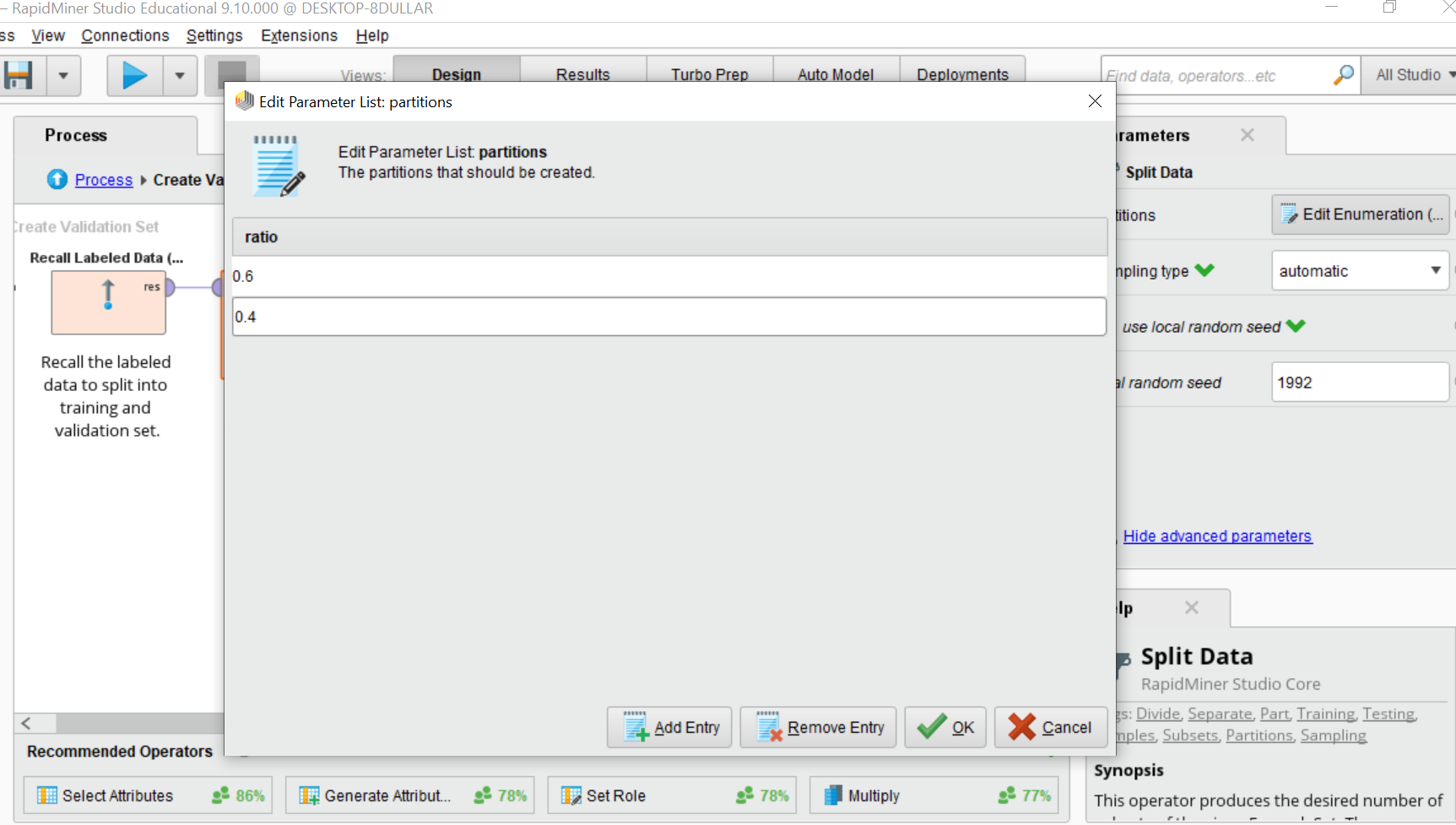
* The labeled data is recalled for splitting into a training and a validation set.
* Split Data - we took the loaded data as our input and delivered the subsets of that data into training data and validation data.

We used stratified sampling. This method of sampling builds random subsets and ensures that the class distribution in the subsets is the same as in the entirety of the loaded data. For example, in the case of a binomial classification, stratified sampling builds random subsets such that each subset contains roughly the same proportions of the two values of the class labels.

The relative size of each partition was set to be 0.6 and 0.4 respectively which was automatically selected as the optimal proportion by Rapid Miner (we had the option of changing it but, following the reference below (Peltonen, 2020), we chose to select this split).

REF: Peltonen, E., Sharmila, P., Asare, K. O., Visuri, A., Lagerspetz, E., & Ferreira, D. (2020). When phones get personal: Predicting Big Five personality traits from application usage. *Pervasive and Mobile Computing*, *69*, 101269.

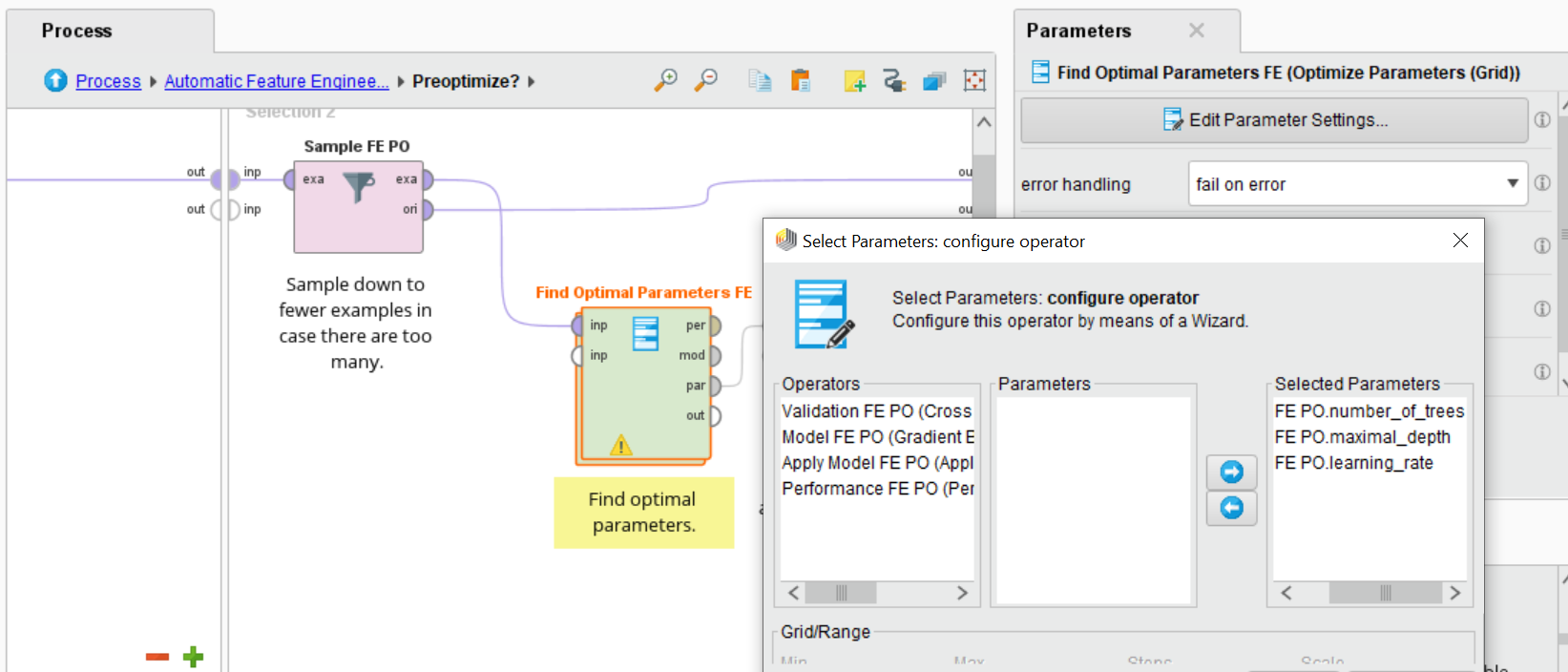




**Automatic Feature Engineering (procedure):**

Perform automatic feature engineering (for training data only):

* Training data was recalled.
* **Pre-Optimize.** We performed a light-weight parameter optimization before we started with the automatic feature engineering to make sure the model is reasonably appropriate.
* Optimize Parameters (Grid). We found the optimal values of the selected parameters for the operators in its subprocess.
* Validate Preoptimize feature engineering. We performed a cross-validation to estimate the statistical performance of the learning model and used the Cross-Validate function for the model to build the final model on complete data. The Cross Validation Operator in RM is a nested operator. It has two subprocesses: a Training subprocess and a Testing subprocess. The Training subprocess was used for training a model. The trained model was then applied in the Testing subprocess. The performance of the model was measured during the Testing phase. The evaluation of the performance of a model on independent test sets yielded a good estimation of the performance on unseen data sets as well as showing any occurrences of “overfitting.”
  + - Number of folds used K=2
    - Sampling type used is shuffled\_sampling. The shuffled sampling builds random subsets of the data. Examples are chosen randomly for making subsets.
    - Local random seed was used.



|  |  |  |  |
| --- | --- | --- | --- |
| Model | Hyperparameter | Value | Scale |
| Gradient Boosted trees | Number of trees | Min: 30  Max: 150  Steps: 2 | Linear |
| Maximal depth | 2, 4 |  |
| Learning rate | Min: 0.01  Max: 0.1  Steps: 1 | Logarithmic |
| Decision Tree | Maximal depth | 2, 7, 15 |  |
| Random Forest | Number of trees | Min: 60  Max: 100  Steps: 1 | Linear |
| Maximal depth | 2, 7 |  |
| Support Vector Machine | Kernel Gamma | Min: 0.05  Max: 5  Steps: 2 | Logarithmic |
| C | Min: 10  Max: 1000  Steps: 1 | Logarithmic |

* Automatic Feature Engineering (procedure):

Performs a fully automated feature engineering process which covers feature selection and feature generation. This process creates an optimal feature set which is then applied to the complete training data to build the final model. The same feature set is also applied on an independent validation set before the prediction model is applied. The operator used a multi-objective evolutionary algorithm for finding the best feature sets. Each feature set is pareto-optimal with respect to complexity vs. model error. The complexity is calculated based on the feature set, where each feature in the set contributes complexity one. The same applies for additional function applications in case of feature generation. The error rate was measured by the performance calculation delivered by the inner operators.

More information about ‘Automatic Feature Engineering’ used in the current research can be found at: **<https://rapidminer.com/webinars-videos/automatic-feature-engineering/>**

* Remember training data.

**Train Model (procedure):**

* Recall training data.
* Optimize Parameters (Grid). We found the optimal values of the selected parameters.
* Validate optimize feature engineering. We performed a cross-validation to estimate the statistical performance of a learning model. We used Cross-validate on the model and built the final model on complete data.
  + Number of folds used K=3.
  + Sampling type: shuffled\_sampling (see description above)
  + Local random seed was used.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Hyperparameter | Value | Scale |
| Gradient Boosted trees | Number of trees | Min: 30  Max: 150  Steps: 2 | Linear |
| Maximal depth | 2, 4, 7 |  |
| Learning rate | Min: 0.01  Max: 0.1  Steps: 2 | Logarithmic |
| Decision Tree | Maximal depth | 2, 4, 7, 10, 15, 25 |  |
| Random Forest | Number of trees | Min: 20  Max: 140  Steps: 3 | Linear |
| Maximal depth | 2, 4, 7 |  |
| Support Vector Machine | Kernel Gamma | Min: 0.05  Max: 5  Steps: 3 | Logarithmic |
| C | Min: 10  Max: 1000  Steps: 2 | Logarithmic |

* Remember training data.

**Transform Validation Data:** Transform the validation data (known target value) using the same preprocessing and features.

* Validation data was recalled.
* The optimal feature set was recalled.
* Apply feature set. The resulting feature set was applied on validation data.
* Remember validation data.

**Create Predictions and Explanations:** Apply the model on the validation and the scoring data sets for scoring. Also explain the predictions and calculated model-specific weights.

* Recalled model data.
* Recalled training data.
* Recalled validation data.
* Explained Predictions using the Rapid Miner operators - which of the attributes plays the largest role in forming that prediction? Calculate model-agnostic global attribute weights. Model-agnostic means that the weights can be calculated for all model types while other model-specific weight calculations only work for particular models (like Random Forest). Model-specific means that the weights are calculated specifically for this model instead of using model-independent weighting schemes like correlations. The operator derives those weights directly from the explanations. If the true labels are known for the test data, all supporting local explanations increase the weights for correct predictions. All contradicting local explanations increase the weights for wrong predictions.

**Validate Model:** Performed a multiple hold-out set validation with robust estimation which provides similar quality of performance estimations to a cross validation with smaller runtimes.

* Recalled model data.
* Recalled validation data.
* Generate Batch - Created index for multiple hold-out sets, dividing the data into batches of the same size. This operator generates a new special batch column which divides the data into a specified number of batches. Those batches are assigned by using the mod function on the row number. Counting starts with 1.

Number of batches = 7

* Performance for Hold-Out sets: Calculate performance for each hold-out set:
  + Apply Model
  + Performance Calculate error on current hold-out set.
* Performance Average (Robust): This operator calculates the average of the input performance vectors. Before doing so, it removes the performances with the highest and the lowest value for the main criterion, making it less likely that outliers will have too great an influence on the average.
* Remember Performance.

**Create Production Model:** Created a final production model by training the model with the same parameters on the combined training and validation data sets.

* Recalled training data.
* Recalled validation data.
* Append (Robust) Appended the training and the validation data, ensuring that the data sets are compatible (same features and nominal values). Like the regular Append operator, this operator adds all the rows of all sample sets into one merged set. However, this operator also keeps all nominal values in place even if they no longer exist in all the data sets. This can be useful in ensuring that models do not break in production, since the metadata of the resulting data does not change in this case.
* Remember Production Data - Remember the data used to train the final production model.
* Build Production Model - Built the final model
  + Recall optimal parameters.
  + Use optimal parameters for the production model.
  + Execute the ML algorithm
* Remember Production Model

**Where to find it on GitHub?**

The 'RapidMiner files' folder contains input files for the RapidMiner Auto-Model (Output from stage 1 - ‘Data\_processing\_output’ folder), as well as processes files (XML files).

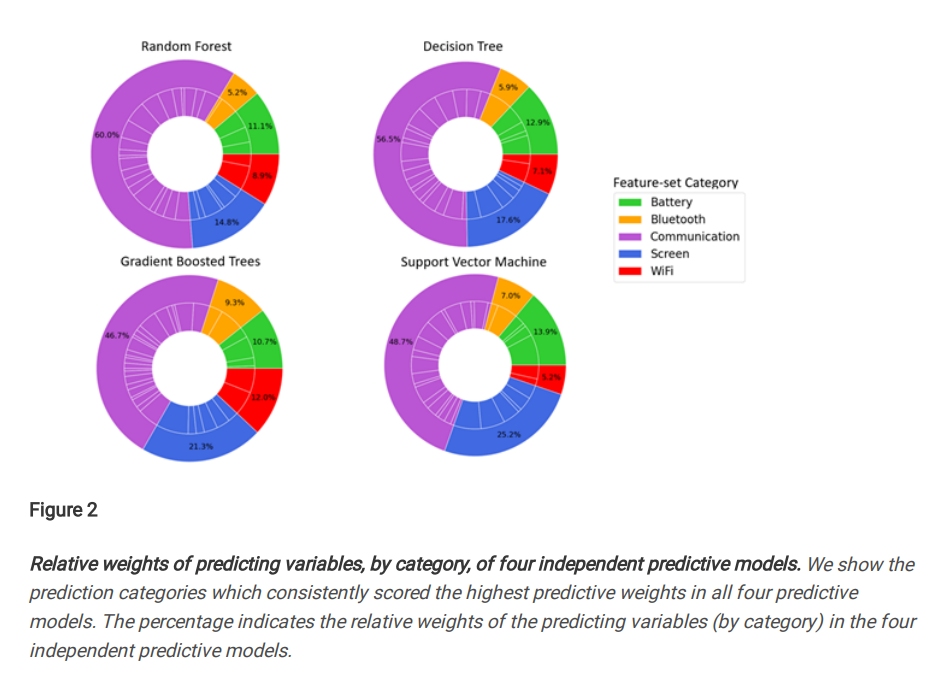
<https://github.com/Mayahocherman/Digital-Phenotyping-of-Smartphone-Data-Successfully-Predicts-a-Broad-Range-of-Personality-Constructs/tree/main/RapidMiner%20files>

# Appendix 1: Feature group importance computation explained (including syntax).

## Background:

Figure 2. in the paper presents the relative weights of predictive variables (organized by category) in each independent predictive model. As outlined in the paper (Methods section), sensor and log data from smartphones - the independent predictive digital footprint variables - were aggregated into five broad functional categories: communication (e.g. average call duration, average number of users a participant has called, average occurrences of incoming messages); screen (e.g. average duration of screen in “On” state, count of occurrences in which the screen was in “On'' state for less than 15 seconds); networking-Wi-Fi (e.g. count of Wi-Fi traces); networking-Bluetooth (e.g. count of Bluetooth traces); and power-battery (e.g. average charges, average occurrences in which the battery state is lower than 20%).

Figure 2 is intended to show which of these categories consistently scored the highest predictive weight in all four predictive models. Following Stachl et al. (2020), Figure 2 illustrates the relative weights of the predictive variables by category in four independent predictive models. Note that all feature categories extracted from our data had some impact on the prediction of a given personality construct, regardless of the specific predictive algorithm utilized. The results show a somewhat similar feature category distribution for all four predictive models. The top five most predictive categories enable us to better understand which feature set is the most applicable for personality prediction and profiling in terms of its consistent predictive value and overall weight, regardless of the specific predictive algorithm used. Note that our results clearly show that the communication feature set, which includes twenty-four sub-features derived from all call and message logs, is the most highly predictive of the majority of personality traits. Put differently, communication data from smartphones is the best overall predictor (in the current study) of a given personality construct. We explain why this is the case in the Discussion section of the paper.



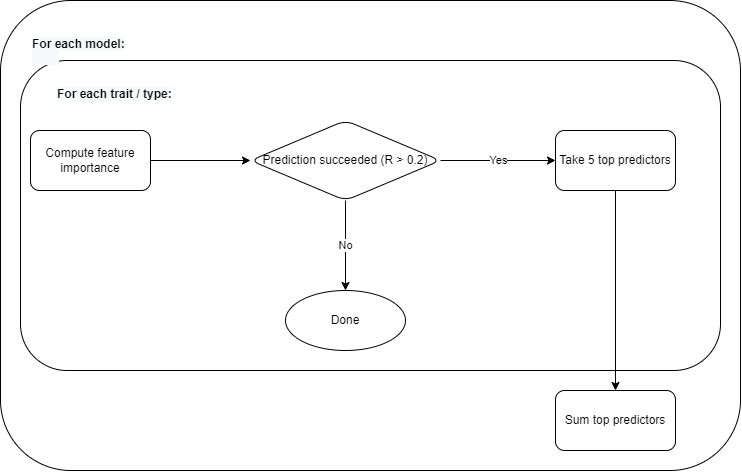
***Figure 2: Relative weights of predictive variables, by category, of four independent predictive models.*** *We show the prediction categories which consistently scored the highest predictive weights**in all four predictive models. The percentage indicates the relative weights of the predictive variables (by category) in the four independent predictive models.*

## Computation method of relative weights:

The weights of each individual feature for each personality construct (taken from the RapidMiner (RM) model output) were evaluated against the following criteria (prediction success of R ≥ 0.2 as explained in the paper). If a feature does not meet the criteria, we do not select it. If it does, we take the top five predictive features and sum them up.

The code is available here starting in line 334:

<https://github.com/Mayahocherman/Digital-Phenotyping-of-Smartphone-Data-Successfully-Predicts-a-Broad-Range-of-Personality-Constructs/blob/main/Visualization.py> Also note that RapidMiner weights for the tree models using the open source [H2O.AI](https://h2o.ai/). To calculate the weights for the SVM model, RapidMiner algorithm uses the internal Java implementation of the [mySVM by Stefan Ruepin](http://www.stefan-rueping.de/publications/rueping-2001-b.pdf).



**Where to find it on GIT?**

The 'RapidMiner files' folder contains input files for the RapidMiner Auto-Model (Output from stage 1 - ‘Data\_processing\_output’ folder), as well as processes files (XML files).

<https://github.com/Mayahocherman/Digital-Phenotyping-of-Smartphone-Data-Successfully-Predicts-a-Broad-Range-of-Personality-Constructs/tree/main/RapidMiner%20files>

# Appendix 2: A comment on small scale Elastic Net data analysis

An Elastic Net (a mix of ridge and lasso regression) prediction model for selected constructs (Big 5 and Dark Triad compounded - Machiavellism, Narcissism, and Psychopathy) was added to the main analysis in order to add a GLM modeling to the Tree based and SVM which we hope to expand the current study with more data and more modeling (including cluster studies) in the future. Please note in the table below that the differences between the SVM and GLM on the Big 5 and four Dark Triad constructs are relatively tiny:

