A Hybrid Deep Learning-Based Feature Selection Approach for Supporting Early Detection of Long-Term Behavioral Outcomes in Cancer Survivors

Tracy Huang, Chun-Kit Ngan, Yin Ting Cheung, Madelyn Marcotte, Benjamin Cabrera

Submitted to: JMIR Bioinformatics and Biotechnology
on: August 01, 2024

Table of Contents

A Hybrid Deep Learning-Based Feature Selection Approach for Supporting Early Detection of Long-Term Behavioral Outcomes in Cancer Survivors

Tracy Huang[1*]; Chun-Kit Ngan[2*]; Yin Ting Cheung[3]; Madelyn Marcotte[2]; Benjamin Cabrera[4]

[1]Emory University Rollins School of Public Health Atlanta US
[2]Worcester Polytechnic Institute Worcester US
[3]The Chinese University of Hong Kong Hong Kong HK
[4]Arizona State University Tempe US
[*]these authors contributed equally

Corresponding Author:
Tracy Huang
Emory University Rollins School of Public Health
1518 Clifton Rd NE
Atlanta
US

Abstract

Background: The number of cancer survivors is growing, and cancer survivors often suffer from long-term behavioral outcomes due to cancer treatments. There is a need for better computational methods to handle and predict cancer behavioral outcomes so physicians and healthcare providers can implement preventative treatments for cancer survivors.

Objective: The aim of this study is to create a new feature selection algorithm to improve the performance of machine learning classifiers to predict long-term behavioral outcomes in cancer survivors.

Methods: We devise a hybrid deep learning-based feature selection approach to support early detection of long-term behavioral outcomes in cancer survivors. Within a data-driven, clinical-domain guided framework to select the best set of features among cancer treatments, chronic health conditions, socio-environmental factors, we develop a two-stage feature selection algorithm, i.e., a multi-metric, majority-voting filter and a deep drop-out neural network, to dynamically and automatically select the best set of features for each behavioral outcome. We also conduct an experimental case study on existing study data with 102 survivors of acute lymphoblastic leukemia (ALL) (aged 15 to 39 years old at evaluation and > 5 years post-cancer diagnosis) who were treated in a public hospital of Hong Kong. Finally, we design and implement radial charts to illustrate the significance of the selected features on each behavioral outcome to support clinical professionals' future treatment and diagnoses.

Results: In this pilot study, we demonstrate that our approach outperforms the traditional statistical and computation methods, including linear and non-linear feature selectors, for the addressed top-priority behavioral outcomes. Our approach holistically has higher F1, precision, and recall scores compared to existing feature selection methods. The models select several significant clinical and socioenvironmental variables as risk factors associated with the development of behavioral problems in young survivors of ALL.

Conclusions: Our novel feature selection algorithm has potential to improve machine learning classifiers' capability to predict long-term behavioral outcomes in cancer survivors.

(JMIR Preprints 01/08/2024:65001)
DOI: https://doi.org/10.2196/preprints.65001

Preprint Settings

1) Would you like to publish your submitted manuscript as preprint?

✔ **Please make my preprint PDF available to anyone at any time (recommended).**
Please make my preprint PDF available only to logged-in users; I understand that my title and abstract will remain visible to all users.
Only make the preprint title and abstract visible.
No, I do not wish to publish my submitted manuscript as a preprint.

2) If accepted for publication in a JMIR journal, would you like the PDF to be visible to the public?

✓ **Yes, please make my accepted manuscript PDF available to anyone at any time (Recommended).**
Yes, but please make my accepted manuscript PDF available only to logged-in users; I understand that the title and abstract will remain v
Yes, but only make the title and abstract visible (see Important note, above). I understand that if I later pay to participate in <a href="http

Original Manuscript

Original Paper

Authors: Tracy Huang[1], MSPH; Chun-Kit Ngan[2], PhD; Yin Ting Cheung[3], PhD; Madelyn Marcotte[2], MSDS; Benjamin Cabrera[4], BS

Institutions:
[1]Department of Epidemiology, Rollins School of Public Health, Emory University, Atlanta, GA, United States

[2]Data Science Program, School of Arts & Sciences, Worcester Polytechnic Institute, Worcester, MA, United States

[3]School of Pharmacy, Faculty of Medicine, The Chinese University of Hong Kong, Hong Kong SAR, China

[4]School of Mathematical and Statistical Sciences, Arizona State University, Tempe, AZ, United States

A Hybrid Deep Learning-Based Feature Selection Approach for Supporting Early Detection of Long-Term Behavioral Outcomes in Cancer Survivors

Abstract

Background: The number of cancer survivors is growing, and cancer survivors often suffer from long-term behavioral outcomes due to cancer treatments. There is a need for better computational methods to handle and predict cancer behavioral outcomes so physicians and healthcare providers can implement preventative treatments for cancer survivors.
Objective: The aim of this study is to create a new feature selection algorithm to improve the performance of machine learning classifiers to predict long-term behavioral outcomes in cancer survivors.
Methods: We devise a hybrid deep learning-based feature selection approach to support early detection of long-term behavioral outcomes in cancer survivors. Within a data-driven, clinical-domain guided framework to select the best set of features among cancer treatments, chronic health conditions, socio-environmental factors, we develop a two-stage feature selection algorithm, i.e., a multi-metric, majority-voting filter and a deep drop-out neural network, to dynamically and automatically select the best set of features for each behavioral outcome. We also conduct an experimental case study on existing study data with 102 survivors of acute lymphoblastic leukemia (ALL) (aged 15 to 39 years old at evaluation and > 5 years post-cancer diagnosis) who were treated in a public hospital of Hong Kong. Finally, we design and implement radial charts to illustrate the significance of the selected features on each behavioral outcome to support clinical professionals' future treatment and diagnoses.
Results: In this pilot study, we demonstrate that our approach outperforms the traditional statistical and computation methods, including linear and non-linear feature selectors, for the addressed top-priority behavioral outcomes. Our approach holistically has higher F1, precision, and recall scores compared to existing feature selection methods. The models select several significant clinical and socioenvironmental variables as risk factors associated with the development of behavioral problems in young survivors of ALL.

Conclusions: Our novel feature selection algorithm has potential to improve machine learning classifiers' capability to predict long-term behavioral outcomes in cancer survivors.
Keywords: Machine Learning; Data-driven, Clinical Domain-guided Framework; Cancer Survivors; Behavioral Outcome Predictions; Feature Selection; Deep Learning; Neural Networks; Hybrid

Introduction

Background

The number of cancer survivors is increasing globally. The American Cancer Society recently reported that in 2023, 1,958,310 new cancer cases were projected to occur in the United States [1]. Treatment advances have resulted in a dramatic improvement in the survival rates of most cancers, especially in developed countries and regions. However, this growing population of cancer survivors may develop a myriad of treatment-related adverse effects that lead to a compromised health status. Studies have also shown that cancer survivors are more likely than the general population to experience negative long-term behavioral outcomes, such as anxiety, depression, attention problems, and sluggish cognitive tempo, after cancer treatments [2]. Thus, developing an effective approach to identify crucial factors and then detect these negative outcomes in advance is needed so that medical therapists can intervene early and take the appropriate actions and treatments promptly to mitigate adverse effects on cancer survivors.

Current Approaches for Detecting Behavioral Outcomes in Cancer Survivors

Currently, to support the identification of relevant factors and the early detection of those behavioral outcomes for cancer survivors, clinical scientists utilize various statistical analyses to understand the relationship among those behavioral outcomes, cancer treatments, chronic health conditions, and socio-environmental factors [3-5]. Specifically, traditional statistical methods (mainly linear regression analysis) are used to extract predictor variables and then model the relationship between the extracted predictor variables and the behavioral outcomes. This analysis assumes that the behavioral outcomes are for the most part linearly correlated with those predictor variables. However, this assumption may not always hold in this complex and dynamic problem. Furthermore, the predictors for those behavioral outcomes extracted by statistical methods may have weak prediction accuracy, as modeling human behavioral outcomes is challenging due to its multifactorial nature (many predictors, as well as interactions among the predictors affecting the outcome), heterogeneity (differences across individuals), non-linearity of data, multicollinearity (highly correlated variables), class imbalance (few observations of the outcome of interest) and missing data [6, 7]. As a result, this class of linear regressors can only account for a small proportion of variance, with limited usability in a clinical setting. Thus, developing an effective computational methodology that can maximize the use of those data for the purpose of prognostic and predictive behavioral outcomes is highly desirable.

To address the above problems, feature selection (FS) techniques in machine learning (ML) play an important role. FS techniques can be broadly divided into four categories: Filter, Wrapper, Embedded, and Hybrid. Filter methods select features based on their statistical significance to the outcome of interest. Unlike other feature selection methods, such as wrapper and embedded methods, filter methods function independent of any machine learning classifiers. However, filter methods are less accurate than other methods of feature selection such as wrapper methods. Additionally, there is a risk of selecting redundant features when using filter methods that do not consider the correlation between features. Wrapper methods use a greedy search algorithm with a classifier to sequentially add and/or remove features from the classifier in order to maximize the specified scoring metrics,

i.e., precision, recall, and F1 score. The output is the best subset of features that the algorithm found. While wrapper methods are good at classification accuracy, they are not efficient in computation time or complexity. Additionally, there is also a risk of overfitting with wrapper methods, where the classifier is highly trained to generate accurate predictions for the training data only and cannot correctly create generalized predictions for testing data or any novel datasets. Embedded methods utilize qualities from both filter and wrapper methods to perform feature selection during the construction of the machine learning classifiers. The baseline embedded methods that are commonly used are Lasso, Ridge, and ElasticNet. However, to effectively use embedded methods, prior knowledge on feature sets is required. Additionally, embedded methods could pose problems when identifying small feature sets. Hybrid methods combine filter and wrapper methods to take advantage of the benefits each method provides while minimizing their limitations [9]. A filter method first selects a subset of features, which are then input into a wrapper method to further select the best subset of features. Since hybrid methods are a combination of filter and wrapper methods, they inherit problems from both; filter methods may cancel important features and wrapper methods are inefficient in computation time.

Goal of This Study

To bridge the above gaps, we propose a hybrid deep learning-based feature selection approach to support early detection of long-term behavioral outcomes in cancer survivors. Specifically, our contributions are four-fold: (1) devise a data-driven, clinical domain-guided framework to select the best set of features among cancer treatments, chronic health conditions, socio-environmental factors, etc.; (2) develop a two-stage feature selection algorithm, i.e., a multi-metric, majority-voting filter and a deep drop-out neural network, to dynamically and automatically select the best set of features for each behavioral outcome; (3) conduct an experimental case study on our existing study data with 102 survivors of acute lymphoblastic leukemia (aged 15 to 39 years old at evaluation and > 5 years post-cancer diagnosis) who were treated in a public hospital of Hong Kong; and (4) design and implement radial charts to illustrate the significance of the selected features on each behavioral outcome to support clinical professionals' future treatment and diagnoses. In this pilot study, we demonstrate that our approach outperforms the traditional statistical and computation methods, including linear and non-linear feature selectors, for the addressed top-priority behavioral outcomes

Methods

Baseline Feature Selection Methods Review

A total of four baseline feature selection methods were used in the experimental studies as a comparison for our novel feature selection algorithm (Table 1).

Table 1. Summary of baseline feature selection methods

Filter	**Wrapper**	**Embedded**	**Hybrid**
Correlation-based Feature Selection (CFS)	Sequential Forward Selection (SFS)	Lasso	CFS → SFS IG → SFS

			MRMR → SFS
Information Gain (IG)	Sequential Backwards Selection (SBS)	Ridge	CFS → SBS IG → SBS MRMR → SBS
Maximum Relevance - Minimum Redundancy (MRMR)	Stepwise Selection (SS)	ElasticNet	CFS → SS IG → SS MRMR → SS

Filter Methods

Filter methods select features based on their statistical significance to the outcome of interest. Unlike other feature selection methods such as wrapper and embedded methods, filter methods function independent of any machine learning classifiers. To evaluate the performance of existing filter methods, we use Information Gain (IG), Maximum Relevance - Minimum Redundancy (MRMR), and Correlation-Based Feature Selection (CFS) [10]. IG is calculated by comparing the entropy of the dataset before and after a transformation. When IG is used for feature selection, it is called Mutual Information, and works by evaluating the IG of each variable in the context of the target. The MRMR algorithm selects the best *K* features at each iteration that have maximum relevance with respect to the target variable and minimum redundancy with respect to the other features. The CFS algorithm involves splitting the features into subsets based on whether their values are continuous or discrete. It can be used to measure the correlation between features and the target outcomes. For continuous data, Pearson's correlation can be used, and for discrete data, symmetric uncertainty can be used. Symmetric uncertainty is a measure of relevance between features and targets that utilizes Mutual Information [11]. When evaluating the performance of the existing filter methods, we select the top fifteen features that had the highest scores for each of the three approaches. As we discuss, filter methods are less accurate than other methods of feature selection such as wrapper methods. Additionally, there is a risk of selecting redundant features when using filter methods that do not consider the correlation between features.

Wrapper Methods

For binary classification, wrapper methods use a greedy search algorithm with a classifier to sequentially add and/or remove features from the classifier in order to maximize the specified scoring metric, i.e., precision, recall, and F1 score. The output is the best subset of features that the algorithm found. To evaluate existing wrapper methods' performances, we select three commonly implemented wrapper methods: sequential forward, sequential backward, and stepwise selection. (1) Sequential forward selection starts with an empty subset of features and iteratively adds features if adding them improves the specified score, according to the machine learning classifier. The selection terminates when a feature subset of the desired size *k*, where *k* refers to the number of features expected by the domain experts, is reached. (2) Sequential backward selection starts with a full subset of all the features and iteratively removes features if removing them increases the specified score, according to the classifier. The selection also terminates when a feature subset of the desired size *k* is reached. (3) Stepwise selection, also known as bidirectional selection, alternates between forward and backward selection in order to select the best subset of features. In order to implement

the wrapper selection approaches, we utilize the Support Vector Machine (SVM) classifier and use the default scoring metric, accuracy [12]. We also specify that the selection process should terminate when a feature subset of size fifteen is reached. While wrapper methods are good at classification accuracy, they are not efficient in computation time or complexity. Additionally, there is also a risk of overfitting with wrapper methods, where the classifier is highly trained to generate accurate predictions for the training data only and cannot correctly create generalized predictions for testing data or any novel datasets.

Embedded Methods

Embedded methods utilize qualities from both filter and wrapper methods in order to perform feature selection during the construction of the machine learning classifier. The embedded classifiers we use are Lasso, Ridge, and ElasticNet. Lasso (Least Absolute Shrinkage and Selection Operator) regression is a form of linear regression that imposes an L1 regularization penalty in order to identify the features which minimize the prediction error [8]. Lasso works by imposing a constraint on model parameters so that the least relevant coefficients shrink towards zero. This is accomplished by having the sum of the absolute value of regression coefficients be less than a specified value. Similar to Lasso, Ridge regression is another form of linear regression that utilizes an L2 penalty. Instead of using the sum of the absolute values, Ridge regression aims to minimize the sum of squares of the regression parameters multiplied. However, no values are shrunk to zero mathematically [13]. ElasticNet regression merges Lasso and Ridge regression by using the L1 and L2 regularization penalties [14]. Therefore, ElasticNet regression is able to shrink some features to zero like Lasso and shrink other features like Ridge. For each evaluated embedded method, we select the top fifteen most relevant features for each behavioral outcome. The challenge is that to effectively use embedded methods, prior knowledge on feature sets is required. Additionally, embedded methods could pose problems when identifying small feature sets.

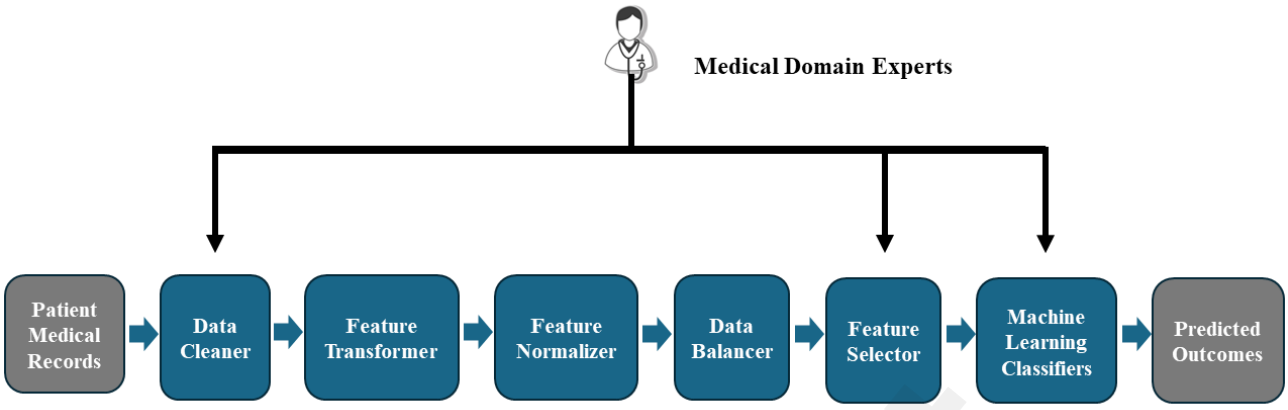
Hybrid Methods

Hybrid methods combine filter and wrapper methods to take advantage of the benefits each method provides while minimizing their limitations [9]. A filter method first selects a subset of features, which are then input into a wrapper method to further select the best subset of features. Since the filter method already minimizes the features the wrapper method needs to choose from, this improves computational time and complexity. At the same time, the wrapper method chooses a more accurate subset from the features chosen from the filter method than if the filter method was utilized by itself. Overall, hybrid methods have improved computational time and accuracy compared to using filter and wrapper methods alone. We implement a total of nine different hybrid methods; we use the top thirty features selected from the three filter methods (i.e., CFS, IG, and MRMR) and input them each into the three wrapper methods, including SFS, SBS, and SS, to subsequently select the top fifteen features. Since hybrid methods are a combination of filter and wrapper methods, they inherit the problems of both, where the filter method may cancel important features and wrapper methods take more time and computational power.

A Data-driven, Clinical Domain-Guided Framework

In this section, we describe and explain our framework that consists of six main modules (Figure 1).

Figure 1. Data-driven, clinical domain-guided framework



The cancer survivor medical records, including the features, such as biomarkers, chronic health conditions, and socio-economic factors, are first passed into the data cleaner that "sanitizes" the records with the clinical domain knowledge from our investigators. Note that throughout the framework, our clinical domain experts assist us with certain processes. In this case study, for example, it consists of replacing missing values in a patient's record by averaging the existing values of the corresponding feature among all the other patients' records grouped by a specific cancer type, age range, and biological sex. Clinical domain experts also help us interpret and explain what different variable values mean for us to properly transform them into the correct variables.

Afterwards, the records are passed into the feature transformer, where the one-hot encoding technique is utilized to transform categorical variables into binary ones [15]. For instance, we transform the "Gender" variable from categorical to binary by replacing "M" and "F" with 1 and 0, respectively.

Following feature transformation, the records are normalized by the feature normalizer. The Shapiro-Wilk Test, the Kolmogorov-Smirnov Test, and the D'Agostino-Pearson Test are utilized to see whether features follow a normal distribution; if two out of the three tests conclude that a feature follows a normal distribution, it is standardized by removing the mean and scaling to unit variance [16-18]. Otherwise, features are normalized using the min-max normalization technique so that all features have values are between "0" and "1"; this will eliminate any feature bias, where features with high values are given more importance than features with low values [19].

Once the records are cleaned, transformed, and normalized, they are then passed into the data balancer. At this point, the results differ depending on the behavioral outcome being predicted. The SMOTE-NC technique is used to artificially balance the instances where the number of patients having a behavioral outcome of "1" is always the minority due to the fact that cancer survivor datasets are often imbalanced [20]. The SMOTE-NC technique is also used to artificially oversample the minority gender so the final datasets can have equal instances of "0" and "1" for the behavioral outcome. For each gender, we specifically chose the SMOTE-NC technique over the regular SMOTE technique because our dataset had a mixture of binary and continuous features. The data is then split into seventy percent training and thirty percent testing data.

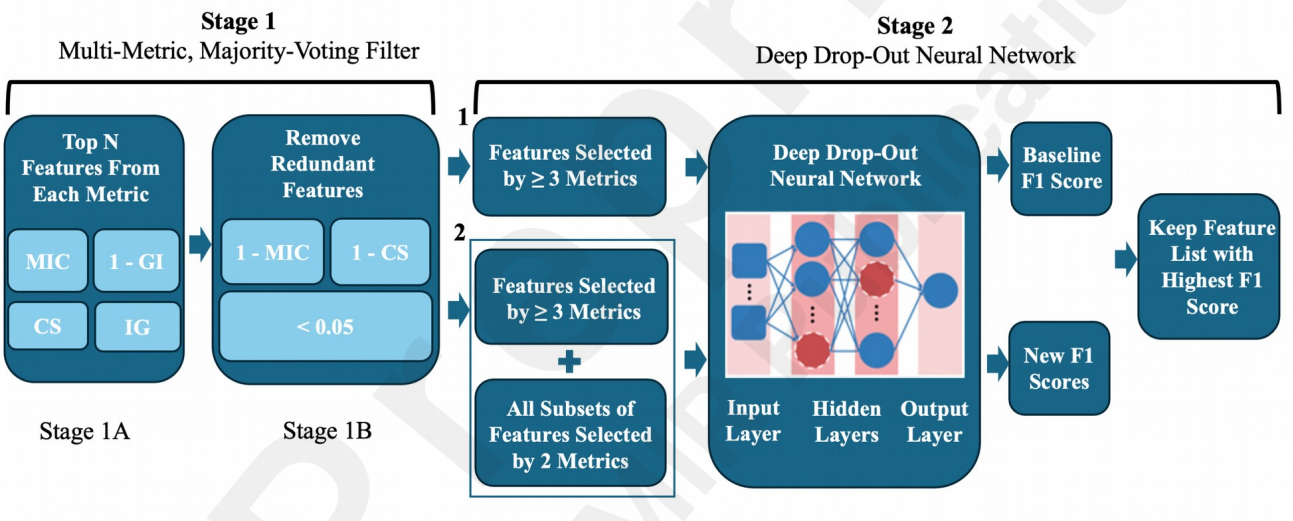
Once the cancer survivor's clinical records pass through all the steps of data pre-processing, they are passed into our hybrid deep learning-based feature selection that is a two-stage feature selection algorithm, i.e., a multi-metric, majority-voting filter and a deep drop-out neural network, to dynamically and automatically select the best set of features for each behavioral outcome. Specifically, the first stage is a novel filter method that utilizes four metrics to select the most

relevant features for a behavioral outcome and removes any redundant features. The second stage is a deep drop-out neural network that replaces a wrapper method, where it further selects features from the ones selected by the multi-metric, majority-voting filter to maximize prediction performance in machine learning classifiers. Note that our clinical domain experts use their clinical expertise to recommend certain features that should be kept in all the final feature lists due to their clinical importance (i.e., gender, current age, and age at diagnosis in our case), if they are not already selected to be in the final feature list by our feature selection approach. Finally, the training data with the final feature list selected from the feature selector with the clinical domain expertise are passed into three machine learning classifiers, including Logistic Regression, Naive Bayes, and K-Nearest Neighbors (kNN), to calculate the precision, recall, and F1 score for the performance evaluation on the testing data.

A Two-stage Feature Selection Algorithm

Our proposed two-stage feature selection algorithm consists of two sequential stages, including a multi-metric, majority-voting filter and a deep drop-out neural network (Figure 2).

Figure 2. Two-stage feature selection algorithm



Stage 1: A Multi-Metric, Majority-Voting (3MV) Filter

Our hybrid deep-learning based feature selection methodology specifically addresses the limitations of existing feature selection methods. In the first stage, it removes redundant features, which some existing filter methods do not consider. Specifically, our 3MV filter has two processing steps in Stage 1.

In Stage 1A, we use four different metrics to select the features that are the most relevant to predict a behavioral outcome. Those metrics include Maximal Information Coefficient (MIC), Gini Index (GI), Information Gain (IG), and Correlation Score (CS) that we calculate between each candidate feature in our pre-processed dataset and the corresponding behavioral outcome of interest, respectively. The MIC is a measure of the strength of the linear or non-linear association between two variables X and Y, where $X \in R$ is the input feature and $Y \in R$ is the corresponding behavioral outcome. The computation is shown in this formula, $MIC(X,Y) = \max_{n_x n_y \le B(n,\alpha)} \left\{ \frac{max_G(I_G(X,Y))}{\log_2(min(n_x,n_y))} \right\}$, where n_x and n_y are the number of bins on the x-axis and y-axis, respectively. G represents a n_x x n_y

grid on (*X*, *Y*), $I_G(X,Y)$ denotes the mutual information under the grid *G*. $B(n,\alpha)$ is a function of data size *n* and is equal to n^{α} ($0 < \alpha < 1$), which limits the maximum number of bins. $\log_2(min(n_x,n_y))$ is a normalization term to ensure *MIC* in the range of 0 to 1. *MIC* converges to 0 as data size $n \rightarrow \infty$ when *X* and *Y* are statistically independent; the *MIC* increases as the correlation between *X* and *Y* strengthens [21].

The GI represents the amount of probability of a specific feature that is classified incorrectly when selected randomly. It is calculated using the formula $GI = 1 - \sum_{i=1}^{n} p_i^2$, where *n* is the number of samples, and p_i is the proportion of the samples that belongs to a distinct behavioral outcome *Y* for the specific feature *X*. Unlike the other three metrics, a higher GI score represents lower associations with the behavioral outcome of interest. To make the scale of the correlation strength between *X* and *Y* consistent among all the metrics, the metric that we use is 1 – GI instead. That is, for all the four metrics, a higher value indicates a higher association with the behavioral outcome of interest.

The IG is a measure of the expected reduction in entropy caused by partitioning the samples according to a specific attribute *X*. That is, $IG(D,X) = E(D) - E(D \vee X)$, where *IG(D, X)* is the information for the dataset *D* for the variable *X*, *E(D)* is the entropy for the dataset with a specific behavioral outcome *Y* before the partitioning, and *E(D|X)* is the conditional entropy for the dataset with a specific behavioral outcome *Y* given the variable *X* [22].

The CS between *X* and *Y* is calculated using the Pearson Correlation Coefficient (PCC), Point Bi-Serial Correlation (PBC), and the PHI-Coefficient (PHI), respectively, based upon the data type of *X* and *Y* [23]. When both *X* and *Y* are the continuous variables, the PCC should be used, which is calculated using the following formula: $PCC = \frac{\sum_{i=1}^{n}(X_i - \acute{X})(Y_i - \acute{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \acute{X})^2 x \sqrt{\sum_{i=1}^{n}(Y_i - \acute{Y})^2}}}$, where X_i are different values of the variable *X*, Y_i are the different value of *Y*, $\acute{X}$ and $\acute{Y}$ are mean values of variable *X* and *Y*, and *n* is the number of samples in the dataset. When comparing one continuous and one binary variable, the PBC is used, calculated with the following formula: $PBC = \frac{\acute{X}_1 - \acute{X}_0}{S_X}\sqrt{\frac{n_0 n_1}{n(n-1)}}$, where $\acute{X}_1$ and $\acute{X}_0$ are the mean values of groups with *Y* = 0 and *Y* = 1, respectively, n_0 and n_1 are the number of samples in a group with *Y* = 0 and *Y* = 1, respectively, n is the total number of samples in the dataset, and $S_X = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \acute{X})^2}$ is the standard deviation of the variable *X* [24]. When comparing two binary variables, the PHI is used, calculated with the following formula: $PHI = \frac{N_{11}N_{00} - N_{10}N_{01}}{\sqrt{N_{X0}N_{X1}N_{Y0}N_{Y1}}}$, where N_{00} = the number of samples in the dataset such that *X* = 0 and *Y* = 0, N_{01} = the number of samples in the dataset such that *X* = 0 and *Y* = 1, N_{10} = the number of samples in the dataset such that *X* = 1 and *Y* = 0, N_{11} = the number of samples in the dataset such that *X* = 1 and *Y* = 1, N_{X0} = the number of samples in the dataset such that *X* = 0, N_{X1} = the number of samples in the dataset such that *X* = 1, N_{Y0} = the number of samples in the dataset such that *Y* = 0, and N_{Y1} = the number of samples in the dataset such that *Y* = 1. All of these measures are values between -1 and 1, with -1 being a perfect negative correlation and 1 being a perfect positive correlation while 0 represents no correlation. We take the absolute value of each measure so that the CS is always

between 0 and 1.

After we calculate the values of all four metrics above between each candidate feature and the behavioral outcome of interest, we rank the top *N* features (which is the number of features expected by the domain experts) for each of the metrics in descending order and store them in a master list, without repetition. From this master list, we construct three feature lists. The first list contains the features selected by at least three metrics, as they are highly likely relevant to predict the behavioral outcome and are then included in the final feature list. The second one contains the features selected by exactly two metrics, as they may be relevant to predict the behavioral outcome and are then needed to be performed the further analysis in Stage 1B. The third one combines all the features from the previous two lists so that we can evaluate the redundancy between any two features from this list.

In Stage 1B, we remove any redundant features from the third combined list generated from Stage 1A. We use the *MIC* and the *CS*; and then calculate these two values for all the feature-to-feature combinations in the combined feature list output from Stage 1A (Albanese et al., 2018). We subtract the *MIC* and the *CS* values from one; and then use the *1 - MIC* and *1 - CS* values to determine if any feature is redundant by other features. The threshold we set is 0.05, based upon our preliminary experimental analysis, so that any combination of two features that results in both scores being less than 0.05 are determined to be redundant. Once it is determined that two features are redundant, we look at the number of metrics that selected the features. If one of the features is selected by fewer metrics, that feature is removed from the third combined list. If both features are selected by the same number of metrics and they are redundant, we then look at the average rank of each feature across the four ranked lists by MIC, GI, IG, and CS, respectively. The feature with the lower rank is removed from the third combined list. The pseudocode algorithm is detailed for the multi-metric, majority-voting (3MV) filter (Textbox 1).

Textbox 1. Step-by-step pseudocode algorithm for multi-metric, majority-voting (3MV) filter

Input:

F: [F_1, F_2, …, F_k], where ***F*** is a set of all input candidate features F_i of the pre-processed clinical records, for $1 \le i \le k$ and $k \in Z^{++}$

***B_Outcome*:** Behavioral Outcome

D_Train: Training Data Set on ***F***

MIC*:** [MIC_1, MIC_2, …, MIC_k], where ***MIC is a set of the corresponding maximal information coefficient values MIC_i between F_i and ***B_Outcome*** computed on ***D_Train***, for $1 \le i \le k$, $MIC_i \in$ ***MIC***, and $k \in Z^{++}$

1-GI*:** [$1\text{-}GI_1$, $1\text{-}GI_2$, …, $1\text{-}GI_k$], where ***1-GI is a set of the corresponding *1-Gini Index$_i$ (GI_i)* values between F_i and ***B_Outcome*** computed on ***D_Train***, for $1 \le i \le k$, $1\text{-}GI_i \in$ ***1-GI***, and $k \in Z^{++}$

CS*:** [CS_1, CS_2, …, CS_k], where ***CS is a set of the corresponding correlation score values CS_i, i.e., Pearson Correlation Coefficient (PCC), Point Bi-Serial Correlation (PBC), or PHI-Coefficient (PHI), between F_i and ***B_Outcome*** computed on ***D_Train*** based upon the data type of F_i and ***B_Outcome***, respectively, for $1 \le i \le k$, $CS_i \in$ ***CS***, and $k \in Z^{++}$

IG*:** [IG_1, IG_2, …, IG_k], where ***IG is a set of the corresponding Information Gain IG_i values between between F_i and ***B_Outcome*** computed on ***D_Train***, for $1 \le i \le k$, $IG_i \in$ ***IG***, and $k \in Z^{++}$

N*:** The Number of Non-redundant Input Candidate Features F_i Expected by Domain Experts, where $F_i \in$ ***F, $N \le k$, and $k \in Z^{++}$

Output:

F*$_{3M+}$**: [T_1, T_2, …, T_p], where ***F*$_{3M+}$** is a set of all non-redundant input candidate features T_j selected by at least three metrics, i.e., MIC_j, $1\text{-}GI_j$, CS_j, and IG_j, for $1 \le j \le p$, $T_j \in$ ***F, $MIC_j \in$ ***MIC***, $1\text{-}GI_j \in$ ***1-GI***, $CS_j \in$ ***CS***, $IG_j \in$ ***IG***, and $p \in Z^{++}$

F*$_{2M}$**: [S_1, S_2, …, S_q], where ***F*$_{2M}$** is a set of all non-redundant input candidate features S_ℓ selected by exactly two metrics, i.e., MIC_ℓ, $1\text{-}GI_\ell$, CS_ℓ, and IG_ℓ, for $1 \le \ell \le q$, $S_\ell \in$ ***F, $MIC_\ell \in$ ***MIC***, $1\text{-}GI_\ell \in$ ***1-GI***, $CS_\ell \in$ ***CS***, $IG_\ell \in$ ***IG***, and $q \in Z^{++}$

Initialization:

3Metrics+ = [] # Store a set of candidate features F_i selected by at least three metrics, i.e., MIC_i, $1\text{-}GI_i$, CS_i, or IG_i, where $1 \le i \le k$, $F_i \in$ ***F***, $MIC_i \in$ ***MIC***, $1\text{-}GI_i \in$ ***1-GI***, $CS_i \in$ ***CS***, $IG_i \in$ ***IG***, and $k \in Z^{++}$

2Metrics = [] # Store a set of candidate features F_i selected by exactly two metrics, i.e., MIC_i, $1\text{-}GI_i$, CS_i, or IG_i, where $1 \le i \le k$, $F_i \in \boldsymbol{F}$, $MIC_i \in \boldsymbol{MIC}$, $1\text{-}GI_i \in \boldsymbol{1\text{-}GI}$, $CS_i \in \boldsymbol{CS}$, $IG_i \in \boldsymbol{IG}$, and $k \in Z^{++}$

3+2Metrics = [] # Store a set of candidate features F_i from both 3Metrics+ and 2Metrics

Rank = [] # Store a set of mean rank positions for each feature in 3+2Metrics. The smaller the position value, the higher the feature rank.

MIC_Feature_Score = [] # Store a set of 1 – MIC[f_i, f_j] values between any pair of two features f_i and f_j in 3+2Metrics computed on ***D_Train***, where $i \neq j$.

CS_Feature_Score = [] # Store a set of 1 – CS[f_i, f_j] values between any pair of two features f_i and f_j in 3+2Metrics computed on ***D_Train***, where $i \neq j$.

M = $\boldsymbol{N}$ # Set the initial number of available input candidate features, where M ≥ $\boldsymbol{N}$ and M ≤ k, for M, $\boldsymbol{N}$, $k \in Z^{++}$

Processing:

STAGE 1A – Select the Top *N* Features Per Metric

***STEP 1*:** Sort F_is in the descending order, according to their MIC_i, $1\text{-}GI_i$, CS_i, and IG_i values, by the developed *sort_features* function and then store their corresponding top M features in the sets, i.e., $\boldsymbol{F_{MIC}}$, $\boldsymbol{F_{1\text{-}GI}}$, $\boldsymbol{F_{CS}}$, and $\boldsymbol{F_{IG}}$, respectively.

$\boldsymbol{F_{MIC}}$ = *sort_features*(***F***, by = ***MIC***, ascending = ***False***).top(M)
$\boldsymbol{F_{1\text{-}GI}}$ = *sort_features*(***F***, by = ***1-GI***, ascending = ***False***).top(M)
$\boldsymbol{F_{CS}}$ = *sort_features*(***F***, by = ***CS***, ascending = ***False***).top(M)
$\boldsymbol{F_{IG}}$ = *sort_features*(***F***, by = ***IG***, ascending = ***False***).top(M)

***STEP 2*:** Create a set $\boldsymbol{F_{UNION}} = \boldsymbol{F_{MIC}} \bigcup_A \boldsymbol{F_{1\text{-}GI}} \bigcup_A \boldsymbol{F_{CS}} \bigcup_A \boldsymbol{F_{IG}}$, where $\bigcup_A$ is a UNION ALL operator that can combine two or more result sets with duplicate values.

***STEP 3*:** Check if a feature $F_i \in \boldsymbol{F}$ appears in at least three metrics in $\boldsymbol{F_{UNION}}$ and then store it in 3Metrics+.

```
for f in F_UNION:
  if COUNT(f) ≥ 3 in F_UNION:
      3Metrics+.add(f)
```

***STEP 4*:** Check if a feature $F_i \in \boldsymbol{F}$ appears in exactly two metrics in $\boldsymbol{F_{UNION}}$ and then store it in 2Metrics.

```
for f in F_UNION:
  if COUNT(f) == 2 in F_UNION:
      2Metrics.add(f)
```

***STEP 5*:** Create a set 3+2Metrics = 3Metrics+ $\bigcup$ 2Metrics, where $\bigcup$ is a UNION operator that can combine two or more result sets without duplicate values.

***STEP 6*:** Calculate the mean ranking position of each feature $F_i \in \boldsymbol{F}$ in 3+2Metrics by the developed *rank* function and then store it in the 1D matrix, i.e., *Rank*.

```
for f in 3+2Metrics:
      r_MIC = rank(f, F_MIC)
      r_1-GI = rank(f, F_1-GI)
      r_CS = rank(f, F_CS)
      r_IG = rank(f, F_IG)
      r_f = ⌈(r_MIC + r_1−GI + r_CS + r_IG) / 4⌉
      Rank[f] = r_f
```

***STEP 7*:** Evaluate if the algorithm has enough input candidate features F_is, expected by domain experts, for the redundancy checking.

```
if size(3Metrics+) > N:
      3Metrics+ = sort_features(3Metrics+, by = Rank, ascending = True)
      del 3Metrics+[N:]
      3+2Metrics = 3Metrics+ ⋃ 2Metrics
      Return 3Metrics+, 2Metrics, and 3+2Metrics
elseif size(3Metrics+) + size(2Metrics) < N:
      M = M + 1
      GoTo STEP 1
else:
      Return 3Metrics+, 2Metrics, and 3+2Metrics
```

STAGE 1B – Remove Redundant Input Features

STEP 1: Compute 1 – MIC[f_i, f_j] values and 1 – CS[f_i, f_j] values, by the developed *compute_MIC* and *compute_CS* functions, respectively, between any pair of two features f_1 and f_2 in 3+2Metrics.

```
for f1 in 3+2Metrics:
    for f2 in 3+2Metrics:
        if f1 ≠ f2:
            MIC[f1, f2] = compute_MIC(f1, f2)
            CS[f1, f2] ] = compute_CS(f1, f2)
            MIC_Feature_Score[f1, f2] = 1 – MIC[f1, f2]
            CS_Feature_Score[f1, f2] = 1 – CS[f1, f2]
```

***STEP 2*:** Iterate each value in MIC_Feature_Score and CS_Feature_Score, respectively, between any pair of two features f_1 and f_2 in 3+2Metrics and then remove the redundant one, i.e., MIC_Feature_Score[f_1, f_2] < 0.05 and CS_Feature_Score[f_1, f_2] < 0.05, according to their counts and ranks in 3Metrics+ and 2Metrics, where 0.05 is the defined threshold.

```
let Temp = 3+2Metrics

for f1 in 3+2Metrics:
    for f2 in 3+2Metrics:
        if f1 ≠ f2 AND MIC_Feature_Score[f1, f2] < 0.05 AND CS_Feature_Score[f1, f2] < 0.05:
            if f1 in 3Metrics+ AND f2 in 2Metrics:
                Temp.remove(f2)
            elseif f2 in 3Metrics+ AND f1 in 2Metrics:
                Temp.remove(f1)
            elseif f1 in 3Metrics+ AND f2 in 3Metrics+:
                if COUNT(f1) in F_UNION > COUNT(f2) in F_UNION:
                    Temp.remove(f2)
                elseif COUNT(f2) in F_UNION > COUNT(f1) in F_UNION:
                    Temp.remove(f1)
                elseif Rank[f1] > Rank[f2]:
                    Temp.remove[f2]
                else:
                    Temp.remove[f1]
            elseif f1 in 2Metrics AND f2 in 2Metrics:
                if Rank[f1] > Rank[f2]:
                    Temp.remove[f2]
                else:
                    Temp.remove[f1]

3+2Metrics = Temp
```

***STEP 3*:** Split 3+2Metrics into two sets, $\boldsymbol{F}_{3M+}$ and $\boldsymbol{F}_{2M}$, respectively.

```
for f in 3+2Metrics:
    if COUNT(f) >= 3 in F_UNION:
        F_3M+.add(f)
    else:
        F_2M.add(f)
```

STEP 4*:** Return $\boldsymbol{F}_{3M+}$ and $\boldsymbol{F}_{2M}$ if the algorithm has enough non-redundant input candidate features F_is, expected by domain experts, or go back to ***STEP 1 of ***STAGE 1A***.

```
if size(F_3M+) + size(F_2M) < N:
    M = M + 1
    GoTo STEP 1 of STAGE 1A
else:
    Return F_3M+ and F_2M
```

For the purpose of illustration, we use our dataset as an example to explain our 3MV filter.

Stage 1A: Select the Top **N** Features Per Metric

Suppose we want to select the best features for predicting the behavioral outcome, Thought Problems. This is our **B_Outcome**. ***F*** is the set of all input candidate features F_i in the pre-processed clinical records. We then calculate the MIC, 1 – GI, IG, and CS scores, respectively, for all the candidate features in the pre-processed clinical records and our B_Outcome, Thought Problems. We store these results in four sets, **MIC**, **(1-GI)**, **IG**, and **CS**, respectively. In this example, our domain experts expect 15 non-redundant input candidate features to be selected; thus, **N** is set to 15.

STEP 1: We first sort the input features (F_is) according to their MIC, 1 - GI, CS, and IG scores. Since **N** is 15, we then take the top 15 features with the highest values from the **MIC** set and place them into a separate set, i.e., F_{MIC}. We repeat this with (1 - GI), IG, and CS scores and place the top 15 features into the corresponding sets, i.e., $F_{1\text{-}GI}$, F_{CS}, and F_{IG}. At this point, we have the following features in these sets: F_{MIC}, $F_{1\text{-}GI}$, F_{CS}, and F_{IG}. Since there are 15 features in each set, we have a total of 60 features across all the four sets (Table 2).

Table 2. Total input features sorted by MIC, 1-GI, CS, and IG scores, respectively, in the descending order

F_{MIC}	$F_{1\text{-}GI}$	F_{CS}	F_{IG}
Physical Fatigue	Years of Education	Physical Fatigue	Impulsivity (on CPT Attention Test)
Overall Fatigue	Intrathecal Chemotherapy	Overall Fatigue	Inattentiveness (on CPT Attention Test)
Cognitive Fatigue	Leukemia Risk Group	Cognitive Fatigue	Information Processing Efficiency (on CPT Attention Test)
Family Communication	Intrathecal MTX Dose	Family Communication	Hematopoietic Stem Cell Transplant
Family Concern	Living Space	IV High-Dose MTX	Response Speed Variability (on CPT Attention Test)
IV High-Dose MTX	Physical Activity	Family Concern	Response speed variability (on CPT Attention test)
Sleep Fatigue	Cognitive Fatigue	Sleep Fatigue	Surgery
Physical Activity	Family Communication	Family Conflict	Sustained Attention (on CPT Attention test)
Family Conflict	Physical Fatigue	Parental Control	Physical Fatigue
Parental Control	Family Mutuality	Physical Activity	Overall Fatigue
Family Mutuality	IV High-Dose MTX	Cranial Radiation Therapy	Neurological Complications
Age at Cancer Diagnosis	Sleep Fatigue	Non-Cranial Radiation	Leukemia Risk Group
Intrathecal MTX Dose	Family Conflict	Intrathecal MTX Dose	Living Space
Non-Cranial Radiation	Age at Cancer Diagnosis	Years of Education	Inattentiveness (on CPT Attention Test)
Cranial Radiation Therapy	Age at Evaluation	Family Mutuality	Inflammatory interleukin-7

[a]CPT: Conner's Continuous Performance Test to measure a person's performance in attention, particularly in areas of inattentiveness, impulsivity, variation in response speed, sustained attention and information processing efficiency

STEP 2: We then create a new set F_{UNION}, the union of sets F_{MIC}, $F_{1\text{-}GI}$, F_{CS}, and F_{IG} in **STEP 1**, allowing duplicate values. This set F_{UNION} represents all the features that have the top 15 MIC, 1-GI, IG, and CS scores. At this point, the set F_{UNION} contains 60 total features.

STEP 3: From the set F_{UNION}, we create the subset **3Metrics+** from the features that are stored in at least three of these four sets, F_{MIC}, $F_{1\text{-}GI}$, F_{CS}, and F_{IG}. These features are then selected as one of the top 15 by at least three out of the four metrics, so these are likely to be highly relevant to predict our **B_Outcome**, Thought Problems; and are included in the final feature list. By applying this concept, the subset **3Metrics+** contains 10 features.

STEP 4: From the set F_{UNION}, we also create a subset **2Metrics** from features that are stored in exactly two out of these four sets, F_{MIC}, $F_{1\text{-}GI}$, F_{CS}, and F_{IG}. These features are selected as the top 15 by two out of the four metrics only. Thus, they may be relevant to predict the **B_Outcome**, Thought Problems, but need to be further analyzed in Stage 2 to determine if they should be kept in the final feature list. By applying this concept, the subset **2Metrics** contains eight features only.

STEP 5: We create another set **3+2Metrics**, i.e., the union of the sets **3Metrics+** and **2Metrics**, without the duplicate values. At this point, the set **3+2Metrics** contains 18 features, including 10 in the **3Metrics+** set and eight in the **2Metrics** set (Table 3).

Table 3. Features in the 3+2Metrics and 2Metrics sets

3Metrics+	2Metrics
Physical Fatigue	Leukemia Risk Group
Overall Fatigue	Living Space
Cognitive Fatigue	Family Concern
Family Communication	Cranial Radiation Therapy
Sleep Fatigue	Years of Education
Family Conflict	Family Control
Family Mutuality	Age at Cancer Diagnosis
Physical Activity	Non-Cranial Radiation
IV High-Dose MTX	
Intrathecal MTX Dose	

STEP 6: We also create a 1D matrix, ***Rank***, which stores the average rank position of each feature in **3+2Metrics** from the sets F_{MIC}, $F_{1\text{-}GI}$, F_{CS}, and F_{IG}. For instance, if we consider the feature “Physical Fatigue”, since its position is 1, 9, 1, and 9 in the sets F_{MIC}, $F_{1\text{-}GI}$, F_{CS}, and F_{IG}, respectively, its average position value in ***Rank*** is equal to 5.

STEP 7: Lastly, we evaluate whether there are too many or too little features at this stage. We first evaluate the number of features in **3Metrics+**. Since **3Metrics+** has 10 features, which is less than **N**, there is no need to remove any extra features. We then evaluate the number of features in **3+2Metrics**. Since there are 18 features in **3+2Metrics**, which is greater than **N**, there is no need to go back to **STEP 1** to find at least 15 features. We now have three sets as the outputs: **3Metrics+** with ten features that are selected by at least three metrics, **2Metrics** with eight features selected by exactly two metrics, and **3+2Metrics**, with 18 features that include the features from both **3Metrics+** and **2Metrics.**

Stage 1B: Remove Redundant Input Features

At this step, we want to remove any redundant features from the features that we selected in Stage

1A.

STEP 1: We compute 1 – MIC[f_i, f_j] values and 1 – CS[f_i, f_j] values, by the developed *compute_MIC* and *compute_CS* functions, respectively, between any pair of two features f_1 and f_2 in 3+2Metrics. We store the 1 – MIC[f_i, f_j] values and 1 – CS[f_i, f_j] values in the sets **MIC_Feature_Score** and **CS_Feature_Score,** respectively.

STEP 2: We iterate each value in **MIC_Feature_Score** and **CS_Feature_Score**, respectively, between any pair of two features f_1 and f_2 in **3+2Metrics**, and check if any values are less than 0.05. We then check if there's any feature pair that has values less than 0.05 in both **MIC_Feature_Score** and **CS_Feature_Score.** Suppose we find that the values in **MIC_Feature_Score** and **CS_Feature_Score** that correspond to the feature pair, "Cranial Radiation Therapy" and "Non-Cranial Radiation", are indeed both less than 0.05. We then select the two features "Cranial Radiation Therapy" and "Non-Cranial Radiation" as the feature pair that we need to further analyze, as they are categorized as the redundant features at this step. Suppose that "Cranial Radiation Therapy" and "Non-Cranial Radiation" are both in the set **2Metrics**, meaning that they are both selected by two metrics. According to the algorithm, since they are selected by an equal amount of metrics, we must compare their rankings in ***Rank*** to decide which one to be removed. Suppose that "Non-Cranial Radiation" had a lower rank (or a higher score) compared to "Cranial Radiation Therapy". Thus, we remove "Non-Cranial Radiation" from the set **3+2Metrics**.

STEP 3: After we remove the redundant features from the set **3+2Metrics**, we then split the set **3+2Metrics** into two new sets: $\boldsymbol{F_{3M+}}$ that its non-redundant features are selected by at least three metrics in the set $\boldsymbol{F_{UNION}}$, and $\boldsymbol{F_{2M}}$, that its non-redundant features are selected by exactly two metrics in the set $\boldsymbol{F_{UNION}}$.

STEP 4: We now have two sets: $\boldsymbol{F_{3M+}}$ and $\boldsymbol{F_{2M}}$**.** The set $\boldsymbol{F_{3M+}}$ has 10 features and the set $\boldsymbol{F_{2M}}$ has seven features after we remove "Non-Cranial Radiation" (Table 4).

Table 4. Non-redundant features in the set F_{3M+} and set F_{2M}

F_{3M+}	F_{2M}
Physical Fatigue	Leukemia Risk Group
Overall Fatigue	Living Space
Cognitive Fatigue	Family Concern
Family Communication	Cranial Radiation Therapy
Sleep Fatigue	Years of Education
Family Conflict	Parental Control
Family Mutuality	Age at Cancer Diagnosis
Physical Activity	
IV High-Dose MTX	
Intrathecal MTX Dose	

At this step, we check if the sum of features from $\boldsymbol{F_{3M+}}$ and $\boldsymbol{F_{2M}}$ is less than 25. Since after removing redundant features, we still have 17 features, which is greater than **N** = 15; thus, we do not need to go back to **STEP 1** in **Stage 1A** to find at least 15 features. We can then proceed to **Stage 2**.

Stage 2: A Deep Drop-out Neural (DDN) Network

In the second stage, the deep neural network has a dropout parameter, where neurons are randomly

ignored during construction of the neural network, in order to avoid model overfitting, which is a problem that the existing wrapper methods have. Thus, our methodology is better suited for finding the best features from the high dimension, low sample size dataset. More specifically, after the features are processed by our 3MV filter, we pass all the non-redundant features to the DDN network that is designed to determine whether or not adding any of those features selected by the only two metrics to the list of the features selected by at least three metrics results in a higher F1 score. Note that this step is not conducted if the number of the non-redundant features, that is, those features have been already selected by at least three metrics in Stage 1, have met the domain experts' expectation. Our designed DDN network is a two-hidden- and one-output-layer architecture. Due to the limited number of patients' medical records with many input features, our DDN network is likely to quickly overfit a training dataset. To address this issue, we utilize the grid search algorithm with the *k*-fold cross validation to find the best dropout rate for our network. We also dynamically set the network's hidden layer size by using the formula, i.e., $\lceil \frac{I+O}{2} \rceil$, where I is the number of selected input subset features and O is the number of labels per behavioral outcome [25]. For the remaining network's initialization parameters, default values are used [26]. The goal is to perform the hyperparameter tuning by using the grid search algorithm with the *k*-fold cross validation to obtain the optimal parameters' values, including the dropout rate, all the network's parameters, and the size of each hidden layer [27].

Specifically, the subset of features selected by three or more metrics in Stage 1 is used in building the initial network architecture to produce the baseline *F1* score. This baseline *F1* score tells us how well the network predicts that a cancer survivor will develop the behavioral outcome of interest, using only the features selected by at least three metrics. Afterwards, we want to see whether or not adding any subset of features selected by two metrics will improve the baseline *F1* score. To achieve this, we try different combinations among the features selected by two metrics, add them on top of the features selected by at least three metrics, use all those features to build, train, and optimize our network by using the grid search algorithm with the *k*-fold cross validation to obtain the optimal parameters' values, and then record each new *F1* score. This allows us to compare *F1* scores between the baseline and the baseline plus additional subsets of features. If any of the new *F1* scores are higher than the baseline, then our final feature list is the one that produces the highest *F1* score. If none of the new *F1* scores are higher than the baseline, then our final feature list is simply the baseline features i.e., the features selected by at least three metrics. A step-by-step pseudocode algorithm for our DDN network is detailed (Textbox 2).

Textbox 2. Step-by-step pseudocode algorithm for DNN network

Input:

***N*:** The Number of Non-redundant Input Candidate Features F_i Expected by Domain Experts, where $F_i \in \boldsymbol{F}$, $\boldsymbol{N} \le k$, and $k \in Z^{++}$

$\boldsymbol{F_{3M+}}$: $[T_1, T_2, \ldots, T_p]$, where $\boldsymbol{F_{3M+}}$ is a set of all non-redundant input candidate features T_j selected by at least three metrics, i.e., MIC_j, $1\text{-}GI_j$, CS_j, and IG_j, for $1 \le j \le p$, $T_j \in \boldsymbol{F}$, $MIC_j \in \boldsymbol{MIC}$, $1\text{-}GI_j \in \boldsymbol{1\text{-}GI}$, $CS_j \in \boldsymbol{CS}$, $IG_j \in \boldsymbol{IG}$, and $p \in Z^{++}$

$\boldsymbol{F_{2M}}$: $[S_1, S_2, \ldots, S_q]$, where $\boldsymbol{F_{2M}}$ is a set of all non-redundant input candidate features S_ℓ selected by exactly two metrics, i.e., MIC_ℓ, $1\text{-}GI_\ell$, CS_ℓ, and IG_ℓ, for $1 \le \ell \le q$, $S_\ell \in \boldsymbol{F}$, $MIC_\ell \in \boldsymbol{MIC}$, $1\text{-}GI_\ell \in \boldsymbol{1\text{-}GI}$, $CS_\ell \in \boldsymbol{CS}$, $IG_\ell \in \boldsymbol{IG}$, and $q \in Z^{++}$

***B_Outcome*:** Behavioral Outcome

Drop_Out_Rate: [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9] is a set of fine-tuning dropout rates for building a DDN network.

D_Train: Training Data Set on $\boldsymbol{F_{3M+}}$

$\boldsymbol{Z}$: $[Z_1, Z_2, \ldots Z_n]$, where $\boldsymbol{Z}$ is a set of Z_rs, for Z_r is a possible subset combination of $\boldsymbol{F_{2M}}$, $Z_r \neq \emptyset$, and $|Z_r| \le \boldsymbol{N} - |\boldsymbol{F_{3M+}}|$, $1 \le r \le n$, and $n \in Z^{++}$

$\boldsymbol{M}$: $[\boldsymbol{F_{3M+}} + Z_r]$, where $\boldsymbol{M}$ is a set of $\boldsymbol{F_{3M+}} \bigcup Z_r$, for Z_r is a possible subset combination of $\boldsymbol{F_{2M}}$, $Z_r \neq \emptyset$, and $|Z_r| \le \boldsymbol{N} - |\boldsymbol{F_{3M+}}|$, $1 \le r \le n$, and $n \in Z^{++}$

E_Train: Training Data Sets on $\boldsymbol{M}$

K: The Number of Training Partitions on ***D_Train*** and ***E_Train*** for Performing Cross-Validation (CV)

Output:

Final_Features: A Set of Final Features Selected by the DDN Network for Building Machine Learning Classifiers.

Initialization:

Learning_Rate = 0.001 # The hyperparameter to govern the pace at which the optimizer algorithm updates the weight values of a DDN network.

Epochs = 500 # The hyperparameter to define the number of times that the learning optimizer works through the entire training dataset on a DDN network

Optimizer = "Adam" # The Adam Optimizer is used for training a DDN network

Loss_Function = "Binary Cross Entropy" # The logarithmic loss to track incorrect labeling of the data class by a DDN network and penalize the network if deviations in probability occur in classifying the labels.

Number_Of_Hidden_Layers = 2 # The number of hidden layers in a DDN network

Number_Of_Output_Layer = 1 # The number of output layer in a DDN network

Hidden_Layer_Size = $\lceil \frac{¿F_{3M+¿\vee+1}}{2} \rceil ¿$ # The number of neurons in each hidden layer, where $¿F_{3M+¿\vee¿¿}$ is the number of input features in $F_{3M+¿¿}$

Output_Layer_Size = 1 # The number of neurons in the output layer

Hidden_Layer_Activation_Function = "Relu" # The neuron activation function used in each hidden layer

Output_Layer_Activation_Function = "Sigmoid" # The neuron activation function used in the output layer

Processing:

STEP 1: Find the best dropout rate, by using the *Grid-Search* technique, F1-score, and ***K***-fold cross-validation, on ***D_Train***, F_{3M+}, ***B_Outcome***, and ***Drop_Out_Rate*** of a DDN network constructed by the *create_DDN* function.

```
let maxF1-score = 0
let bestDropOutRate = 0
for dor in Drop_Out_Rate:
    DDN = create_DDN(dor, ¿F_{3M+¿∨¿¿}, Number_Of_Hidden_Layers, Number_Of_Output_Layer, Hidden_Layer_Size,
    Output_Layer_Size, Hidden_Layer_Activation_Function, Output_Layer_Activation_Function)
    F1-score[dor] = DNN.train_model(D_Train, B_Outcome, K, Learning_Rate, Epochs, Optimizer, Loss_Function)
    if F1-score[dor] > maxF1-score:
        maxF1-score = F1-score[dor]
        bestDropOutRate = dor
```

STEP 2: Construct a DDN network, by the *create_DDN* function, on bestDropOutRate, ***D_Train***, F_{3M+}, and ***B_Outcome***, and then perform ***K***-fold cross-validation to obtain the baseline F1 score, i.e., $\mathbf{F1}_{\mathbf{Baseline}}$.

STEP 3: Iterate each feature set [$F_{3M+} + Z_r$] in ***M*** and construct a DDN network, by the *create_DDN* function, on bestDropOutRate, ***E_Train***, [$F_{3M+} + Z_r$], and ***B_Outcome***, and then perform the ***K***-fold cross-validation to obtain its F1 score, i.e., $\mathbf{F1}_r$, where $1 \le r \le n$

Final_Features = $F_{3M+¿¿}$

$\mathbf{F1}_{\mathbf{Max}} = \mathbf{F1}_{\mathbf{Baseline}}$

```
for fs in M:
    Hidden_Layer_Size = ⌈(fs+1)/2⌉
    DDN = create_DDN(bestDropOutRate, |fs|, Number_Of_Hidden_Layers, Number_Of_Output_Layer, Hidden_Layer_Size,
    Output_Layer_Size, Hidden_Layer_Activation_Function, Output_Layer_Activation_Function)
    F1_fs = DNN.train_model(E_Train, B_Outcome, K, Learning_Rate, Epochs, Optimizer, Loss_Function)
    if F1_fs > F1_Max:
        F1_Max = F1_fs
        Final_Features = fs
```

STEP 4: Return ***Final_Features***

Let us use our dataset as an example to explain our DDN network. At this stage, we want to

determine whether any features selected by two metrics should be kept in the final feature list on top of the features selected by at least three metrics. Our inputs include:

(1). $\boldsymbol{F_{3M+}}$ and $\boldsymbol{F_{2M}}$, which are our outputs from Stage 1B.
(2). ***Drop_Out_Rate***, a set of fine-tuning dropout rates for building a DDN network.
(3). ***D_Train***, which is the training dataset that only includes features in $\boldsymbol{F_{3M+}}$.
(4). **Z**, the set that includes all possible subsets from $\boldsymbol{F_{2M}}$, excluding the null set, where the size of subsets is less than or equal to **N** minus the size of $\boldsymbol{F_{3M+}}$ so that the total number of features does not exceed **N**. In our example, the set **Z** only includes all the possible subsets of size five or less because we already have ten features in **non_redundant_three_more** and **N** minus 10 is five. Given that there are seven features in **non_redundant_two,** there are actually 128 possible subsets. However, because we only need the subsets with size less than or equal to five and we also exclude the null set, we end up with a total of 119 different subsets in the set **Z**.
(5). **M**, a set of lists that add all the possible subsets in the set **Z** to the set $\boldsymbol{F_{3M+}}$; thus, there are 119 different lists.
(6). ***E_Train***, which is the set of training datasets that includes features in each list in **M**.
(7). ***K***, the Number of Training Partitions on ***D_Train*** and ***E_Train*** for Performing Cross-Validation (CV)

STEP 1: We want to find the best dropout rate for the neural network, using the *Grid-Search* technique, F1-score, and ***K***-fold cross-validation, on ***D_Train***, $\boldsymbol{F_{3M+}}$, ***B_Outcome***, and ***Drop_Out_Rate*** of a DDN network. ***K*** is set to 5. We thus first construct a neural network using the *create_DDN* function to perform the *Grid-Search* technique. The neural network is initialized to have a learning rate of 0.001, 500 epochs, use the 'Adam' optimizer, use the 'Binary Cross Entropy' loss function, have two hidden layers with $\lceil \frac{¿F_{3M+¿\vee+1}}{2} \rceil ¿$ number of neurons and the "Relu" activation function, and one output layer with one neuron and the activation function "Sigmoid". Suppose using the *Grid-Search* technique with ***D_Train***, $\boldsymbol{F_{3M+}}$, the ***B_Outcome,*** Thought Problems, ***Drop_Out_Rate***, using 5-fold cross-validation, we get that the best dropout rate is 0.1 (***bestDropOutRate*** is set to 0.1).

STEP 2: We construct a deep neural network with the initialized attributes in the *create_DDN* function, ***bestDropOutRate***, ***D_Train,*** $\boldsymbol{F_{3M+}}$, and ***B_Outcome***, and then perform 5-fold cross-validation to obtain the baseline F1 score, $\mathbf{F1_{Baseline}}$.

STEP 3: We then iterate through each feature set $\boldsymbol{[F_{3M+} + Z_r]}$ in ***M*** and construct a deep neural network with the same initialized attributes in the *create_DDN* function, ***bestDropOutRate***, ***E_Train,*** $\boldsymbol{[F_{3M+} + Z_r]}$and ***B_Outcome,*** and then perform 5-fold cross-validation to obtain the F1 score (**F1**) for each training dataset in ***E_Train***. The hidden layer size of each neural network is calculated using the number of features in M + 1, divided by 2. If any **F1** score is greater than $\mathbf{F1_{Baseline}}$, the final feature list (***Final_Features***) is set to the feature set $\boldsymbol{[F_{3M+} + Z_r]}$ in ***M*** in which the **F1** score is obtained.

STEP 4: We have the feature list with the best F1 score (***Final_Features***), which is passed into three machine learning classifiers, Logistic Regression, Naive Bayes, and k-Nearest Neighbors.

Pilot Experimental Study

In our experimental study, we used a 2018-2020 dataset that contains 102 acute lymphoblastic leukemia survivors' clinical records collected from a public hospital in Hong Kong. This study was approved by the CUHK-NTEC Clinical Research Ethics Committee. The survivors were between the age of 15 to 39 years, had completed treatment, and were more than 5 years post-cancer diagnosis at the time of recruitment. In each patient record, there are more than 50 features, including demographic factors (e.g., age, gender, education level, etc.), cancer treatments received (e.g., radiation, chemoradiotherapy, surgery, etc.), inflammatory biomarkers (e.g., IL-7, MCP-1, TNF-α, etc.), physical health conditions (e.g., body mass index, sleep fatigue, and cognitive fatigue, etc.), family life and socioeconomic descriptors (e.g., family conflict, family communication, living space, etc.), attention-related outcomes (e.g., measures of inattentiveness, impulsivity, sustained attention, etc.), and lifestyle habits (e.g., drinking, smoking, physical activity, etc.).

After preprocessing the data and using our two-stage feature selection algorithm, we selected approximately 15 input features, expected by our medical investigators, to train and test our three machine learning classifiers, i.e., Logistic Regression, Naïve Bayes, and K-Nearest Neighbors, respectively, to predict six behavioral outcomes (i.e., Anxiety and Depression, Thought Problems, Attention Problems, Internalizing Problems, Externalizing Problems, and Sluggish Cognitive Tempo) that our medical investigators would like to focus on. Due to their clinical importance recommended by our medical investigators, we also add three more clinically-relevant features (i.e., gender, current age, and age at diagnosis) to the final feature list if those features have not been already selected by our two-stage feature selection approach.

Results

The experimental results include the F1 score, Precision, and Recall on the testing data (Tables 5, 6, and 7, respectively). Note that for each feature selection method category, those scores are the average values of prediction performance among all the three machine learning classifiers for every behavioral outcome. In the "Our Method" column, the italics and underline means that our score is higher than all the four baseline methods. The italics only means that our score is slightly lower than the highest baseline method score.

Table 5. Average F1 Scores

Behavioral Outcome	Filter	Wrapper	Embedded	Hybrid	Our Method	Percent Change (Our Method vs. Highest Baseline)
Anxiety and Depression	0.624	0.437	0.585	0.449	*0.738*	+ 18.27%
Thought Problems	0.490	0.438	0.477	0.394	*0.511*	+ 4.29%
Attention Problems	0.348	0.417	0.440	0.350	*0.568*	+ 29.10%
Internalizing Problems	0.533	0.706	0.619	0.637	*0.700*	- 0.85%
Externalizing Problems	0.219	0.459 SD (0.309)	0.267	0.265	0.278 SD (0.208)	- 39.43%
Sluggish Cognitive Tempo	0.560	0.463	0.582	0.489	*0.639*	+ 9.79%

Table 6. Average Precision Scores

Behavioral Outcome	Filter	Wrapper	Embedded	Hybrid	Our Method	Percent Change (Our Method vs. Highest Baseline)
Anxiety and Depression	0.562	0.407	0.563	0.424	*0.708*	+ 25.75%
Thought Problems	0.522	0.385	0.590 SD (0.140)	0.496	0.448 SD (0.041)	- 24.07%
Attention Problems	0.290	0.360	0.350	0.329	*0.515*	+ 43.10%
Internalizing Problems	0.583	0.665	0.665	0.668	*0.618*	- 7.49%
Externalizing Problems	0.230	0.417	0.278	0.297	*0.444*	+ 6.47%
Sluggish Cognitive Tempo	0.542	0.409	0.577	0.494	*0.570*	- 1.21%

Table 7. Average Recall Scores

Behavioral Outcome	Filter	Wrapper	Embedded	Hybrid	Our Method	Percent Change (Our Method vs. Highest Baseline)
Anxiety and Depression	0.813 SD (0.098)	0.519	0.630	0.580	0.778 SD (0.079)	- 4.31%
Thought Problems	0.537	0.556	0.463	0.383	*0.611*	+ 9.89%
Attention Problems	0.463	0.519	0.630	0.424	*0.667*	+ 5.87%
Internalizing Problems	0.587	0.762	0.651	0.651	*0.857*	+ 12.47%
Externalizing Problems	0.222	0.556 SD (0.416)	0.259	0.259	0.222 SD (0.157)	- 60.07%
Sluggish Cognitive Tempo	0.654	0.568	0.617	0.568	*0.741*	+ 13.30%

Our two-stage feature selection approach outperforms or levels the existing feature selection methods to support the prediction of five out of six behavioral outcomes (i.e., Anxiety and Depression, Thought Problems, Attention Problems, Internalizing Problems, and Sluggish Cognitive Tempo) in terms of the average F1 scores (Table 5). Although the wrapper method outperforms our feature selection approach to support the prediction of Externalizing Problems, our approach's performance is more stable, as the F1 score variance is smaller. Thus, our feature selection approach still outperforms the other three existing feature selection methods.

Additionally, our feature selection approach outperforms or levels the existing feature selection methods to support the prediction of five out of six behavioral outcomes (i.e., Anxiety and Depression, Attention Problems, Internalizing Problems, Externalizing Problems, and Sluggish Cognitive Tempo) in terms of precision scores (Table 6). Although the embedded method

outperforms our feature selection approach to support the prediction of Thought Problems, our approach's performance variance is much smaller that implies our approach is more stable.

Lastly, our feature selection approach outperforms the existing feature selection methods to support the prediction of four out of six behavioral outcomes (i.e., Thought Problems, Attention Problems, Internalizing Problems, and Sluggish Cognitive Tempo) in terms of recall scores (Table 7). Although the filter and wrapper method outperform our feature selection approach to support the prediction of Anxiety and Depression and Externalizing Problems, respectively, our approach's performance variance is much smaller as well.

Since the F1 scores are calculated from both precision and recall scores, we can infer that our feature selection approach improves the F1 scores largely because it increases the recall scores as opposed to the precision scores (Tables 6 and 7). Overall, the experimental results show promising evidence that our method improves the ML classifiers' prediction performance to support better early detection of long-term behavioral outcomes in cancer survivors.

Radial Feature Charts

Radial feature charts are generated for each of the six behavioral outcomes analyzed, including Anxiety and Depression, Thought Problems, Attention Problems, Internalizing Problems, Externalizing Problems, and Sluggish Cognitive Tempo (Figure 3 - 8). Each chart includes the top 15+ features selected by our proposed methodology. The size and the color of each red slice is measured by the Unified Metric Value of each feature, which is calculated by averaging the scores of the metrics that select each feature during Stage 1A of our proposed method.

Figure 3. Anxiety and Depression Radial Feature Chart.

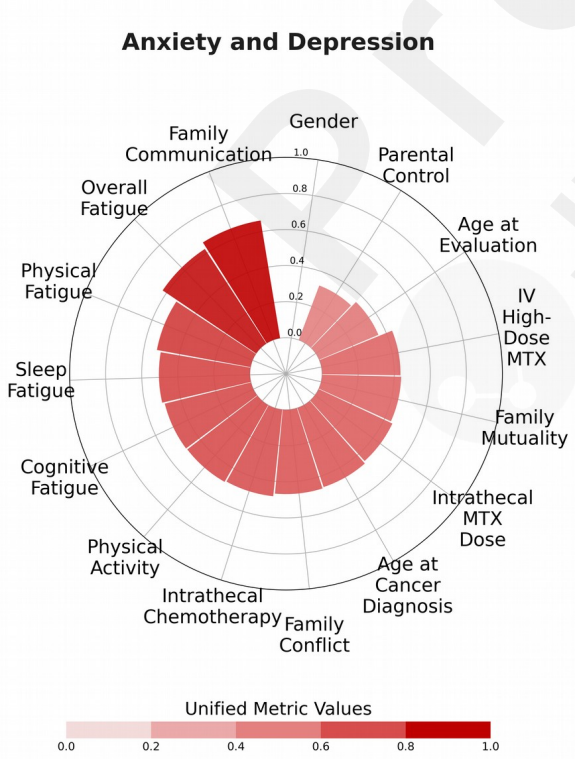


Figure 4: Sluggish Cognitive Tempo Radial Feature Chart.

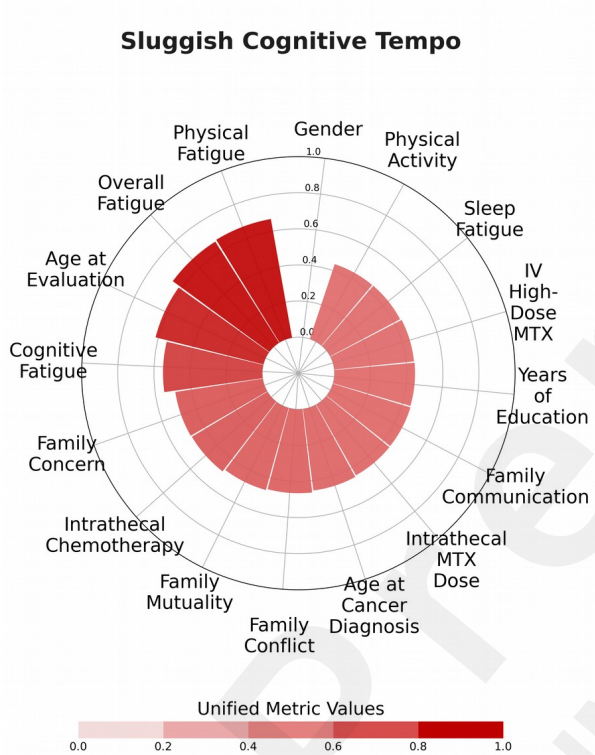


Figure 5: Externalizing Problems Radial Feature Chart.

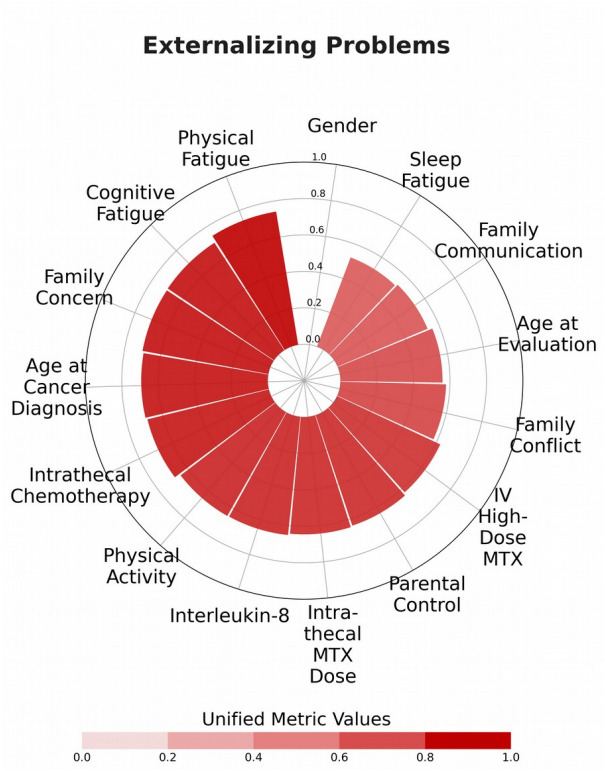


Figure 6: Internalizing Problems Radial Feature Chart.

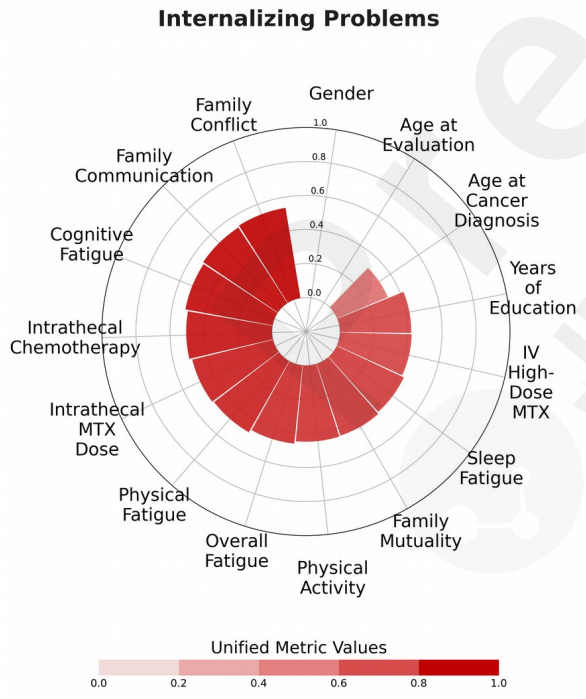


Figure 7: Attention Problems Radial Feature Chart

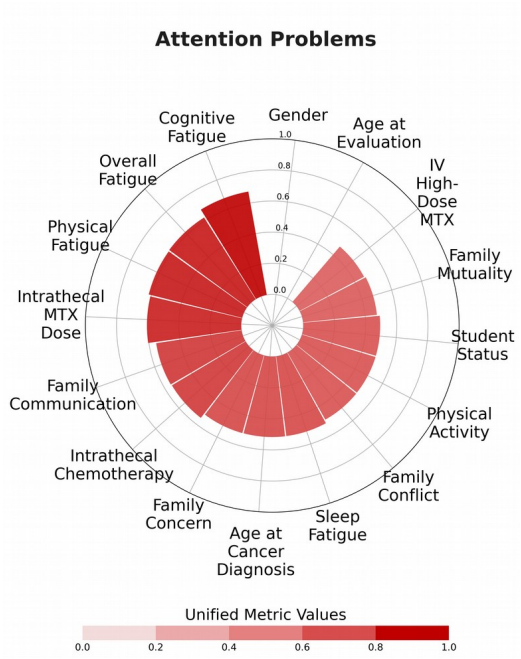
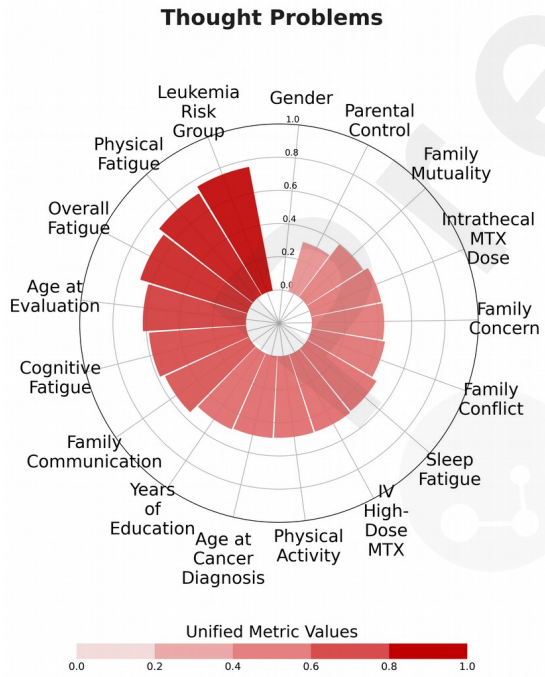


Figure 8: Thought Problems Radial Feature Chart.



[a]The variables represent the documented risk factors associated with the development of behavioral problems in the literature. They include (1) sociodemographic variables (age at evaluation and gender), (2) clinical variables (age at cancer diagnosis, intrathecal chemotherapy, intrathecal methotrexate dose, IV high-dose methotrexate, and inflammatory interleukin-8 levels), (3) socioenvironmental and lifestyle variables (sleep, fatigue, physical activity, and family functioning). Physicians can interpret the charts by seeing which features have the darkest color and largest size, indicating higher Unified Metric Values and thus greater associations with the behavioral problem of

interest. Those features can then be further used to devise customized prevention plans and advice.

Discussion

Principal Results

In this work, we sought to develop a prognostic machine learning framework and feature selection approach to predict functional outcomes trajectory in the specific population of ALL cancer survivors. Our hybrid deep learning-based feature selection approach out-performs and/or levels the existing feature selection methods assessed (filter, wrapper, embedded, and hybrid) for five out of six long-term behavioral outcomes. Even in cases where our feature selection method did not outperform existing methods, our approach's performance variance was much smaller and thus more stable. From the data, we infer that our feature selection approach improves F1 scores from machine learning classifiers compared to existing feature selection methods largely because it increases the recall scores as opposed to the precision scores. We also developed radial feature charts that can quickly and effectively help clinicians understand which predictor variables were most important in predicting long-term behavioral outcomes. Overall, the experimental results show promising evidence that our method improves ML classifiers' prediction performance on high dimension low sample size data, which can support better early detection of long-term behavioral outcomes in cancer survivors.

Limitations

Our study was limited to a pilot study with young Chinese survivors of leukemia, which limits the generalizability of the results. Future studies should utilize our feature selection approach on different demographics and types of cancer survivors. Additionally, even though clinical domain experts assisted with additional input for the features that are kept in machine learning classifiers, there remains room for human error and domain experts' opinions may occasionally differ from what features would optimize machine learning classifiers' performance. Lastly, additional biases may have influenced the data, such as biases in patients who were able to access hospital care and were willing to have their data shared with our clinical investigators.

Comparison with Prior Work

Our findings reinforce existing evidence that behavioural outcomes in cancer survivors are a complex and multifactorial phenotype. Most pre-existing research is focused on either disease- or treatment-related factors as predictors of cognitive dysfunction. However, socio-environmental factors play an important role in the neurodevelopment of these young survivors. Our findings showed the interaction and unique contribution of the socio-environmental factors, such as family dynamics and lifestyle factors, on anxiety, depressive and sluggish-cognitive symptoms in survivors. Studies have found associations of parents' psychological distress on the child's cognitive and behavioral outcomes [4, 28]. Environmental events can elicit a biological stress response that results in neurological reactions to that stress. This is especially relevant in the context of Hong Kong and Mainland China where much emphasis is now placed on ameliorating the adverse health effects of the urban environment in children and adolescents. The findings provide directions for the development of multidisciplinary services and interventions. For example, social workers can pay more attention to the occupational/ employment challenges of young survivors who experience fatigue symptoms from treatment or/and manifest adverse behavioral outcomes. The study findings can help us identify high-risk subgroups from dysfunctional families or households struggling with financial problems and conflicts. Interventions that promote self-confidence and positive peer interaction can be implemented during the early survivorship phase when young survivors transit back to their full-time school or work.

Our results also build upon existing computational methods and feature selection approaches for predicting behavioral outcomes in cancer survivors. Traditional computational methods in the clinical and social sciences typically employ the use of regression analysis to model the relationship between two or more variables for prediction. However, modeling human behavioral data is challenging due to issues such as its multifactorial nature, heterogeneity, non-linearity of data, and class imbalance [6, 7]. As a result, the model can only account for a small proportion of variance with limited utility in clinical settings. For example, we have reported that cranial radiation, chronic health conditions, and poor physical activity are associated with worse cognitive and behavioral outcomes in Chinese survivors of childhood leukemia [5]**.** However, these factors only accounted for 22.9% to 35.8% of the variance in the traditional regression models. Identifying an effective computational method that minimizes algorithmic bias, like the two -stage feature selection algorithm within the clinical domain-guided framework outlined in this study, can maximize the use of clinical and behavioral data for predictive purposes. Such prognostic models will aid in informing strategies aimed at changing behavior and designing social and clinical interventions.

Conclusions

Given that we are working with such small cancer survivor datasets, even a slight improvement in prediction performance from machine learning classifiers can make a big difference in helping cancer survivors. Our data-driven, clinical domain-guided approach can potentially address the problem of "High-Dimension Low-Sample Size"; the pilot analysis shows that this approach has allowed us to identify a set of interacting clinical and socio-environmental characteristics that predicted behavioral outcomes in survivors.

In late 2019, the American Cancer Society had a special call for attention to financial, social and emotional concerns that uniquely affect young cancer survivors [29]. Currently in Hong Kong, there is no centralized cancer programs for adolescent and young adult patients. From a clinical perspective, identifying the unique factors associated with inter-individual differences in functional outcomes will help clinicians to identify individualized modifiable risk factors; this will contribute to the development of a personalized, patient-centered cancer care program for local cancer patients. From a research perspective, this project serves as a pilot study to apply machine-learning based prognostic technology, guided by clinical knowledge, on a combination of objective data (clinical and demographics variables) and subjective data (behavioral and patient-reported variables). The framework and algorithms developed through this analysis can be applied to address clinically relevant research questions in patients with other chronic diseases. The aim of this application is in line with the recent call by the Government of the Hong Kong Special Administrative Region to harness data-driven analytics to formulate healthcare policies [30].

Acknowledgements

This research study is supported by the U.S. National Science Foundation (ref no: 1852498) awarded to Chun-Kit Ngan and partially supported by the Hong Kong Research Grant Council Early Career Scheme (ref no: 24614818) and General Research Fund (ref no: 14604022) awarded to Yin Ting Cheung.

Conflicts of Interest

None declared.

Abbreviations

3MV: multi-metric majority-voting
ALL: acute lymphocytic leukemia
CS: correlation score
CFS: correlation-based feature selection
DDN: deep neural network
FS: feature selection
GI: gini index
Lasso: least absolute shrinkage and selection operator
IG: information gain
kNN: k-nearest neighbor
MIC: maximal information coefficient
ML: machine learning
MRMR: maximum relevance – minimum redundancy
PBC: point bi-serial correlation
PCC: pearson correlation coefficient
PHI: PHI-coefficient
SFS: sequential forward selection
SBS: sequential backwards selection
SMOTE: synthetic minority oversampling technique
SMOTE-NC: synthetic minority over-sampling technique for nominal and continuous
SS: stepwise selection
SVM: support vector machine

References

1. Siegel RL, Miller KD, Wagle NS, Jemal A. Cancer statistics, 2023. Ca Cancer J Clin. 2023;73(1):17-48. doi: 10.3322/caac.21442
2. Brinkman TM, Recklitis CJ, Michel G, Grootenhuis MA, Klosky JL. Psychological symptoms, social outcomes, socioeconomic attainment, and health behaviors among survivors of childhood cancer: current state of the literature. Journal of Clinical Oncology. 2018;36(21):2190. doi: 10.1200/JCO.2017.76.5552
3. Alias H, Morthy SK, Zakaria SZS, Muda Z, Tamil AM. Behavioral outcome among survivors of childhood brain tumor: a case control study. BMC pediatrics. 2020;20:1-10. PMID: 32020861
4. Patel SK, Wong AL, Cuevas M, Van Horn H. Parenting stress and neurocognitive late effects in childhood cancer survivors. Psycho-Oncology. 2013;22(8):1774-82. doi: 10.1002/pon.3213
5. Peng L, Yang LS, Yam P, Lam CS, Chan AS-y, Li CK, Cheung YT. Neurocognitive and behavioral outcomes of Chinese survivors of childhood lymphoblastic leukemia. Frontiers in oncology. 2021;11:655669. PMID: 33959507
6. Kliegr T, Bahník Š, Fürnkranz J. Advances in machine learning for the behavioral sciences. American Behavioral Scientist. 2020;64(2):145-75. doi: 10.48550/arXiv.1911.03249
7. Turgeon S, Lanovaz MJ. Tutorial: Applying machine learning in behavioral research. Perspectives on Behavior Science. 2020;43(4):697-723. doi: 10.1007/s40614-020-00270-y
8. Jonas R, Cook J. Lasso regression. British Journal of Surgery. 2018;105(10). doi: 10.1002/bjs.10895
9. Thejas G, Garg R, Iyengar SS, Sunitha N, Badrinath P, Chennupati S. Metric and accuracy ranked feature inclusion: Hybrids of filter and wrapper feature selection approaches. IEEE Access. 2021;9:128687-701. doi: 10.1109/ACCESS.2017.DOI
10. Cherrington M, Thabtah F, Lu J, Xu Q, editors. Feature selection: filter methods performance

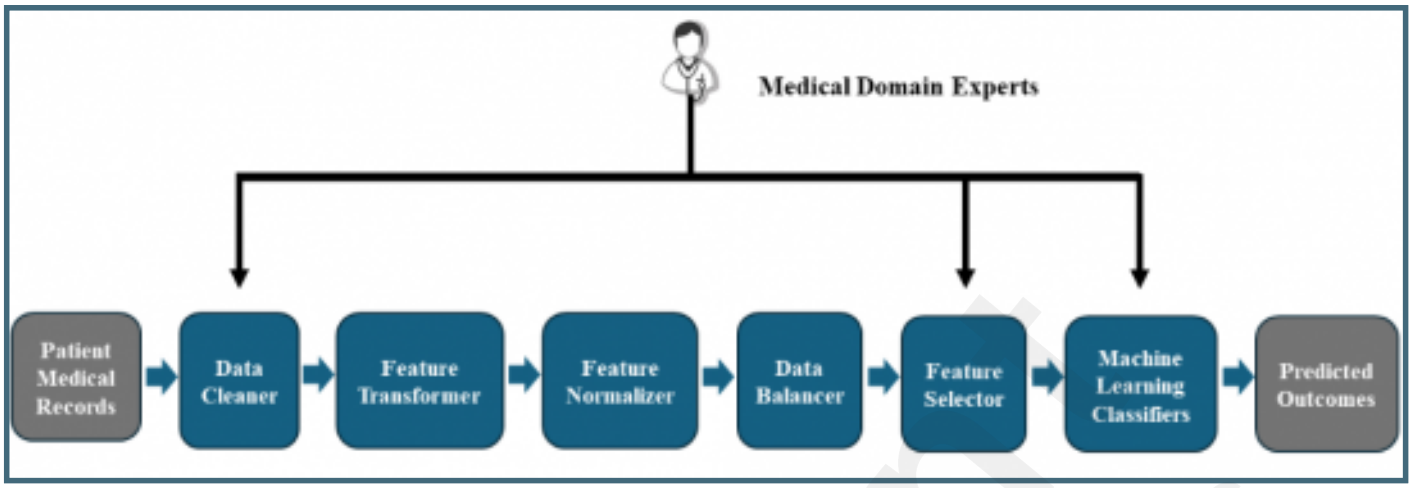
challenges. 2019 International Conference on Computer and Information Sciences (ICCIS); 2019: IEEE. doi: 10.1109/ICCISci.2019.8716478
11. Lin X, Li C, Ren W, Luo X, Qi Y. A new feature selection method based on symmetrical uncertainty and interaction gain. Computational biology and chemistry. 2019;83:107149. PMID: 31751882
12. Huang S, Cai N, Pacheco PP, Narrandes S, Wang Y, Xu W. Applications of support vector machine (SVM) learning in cancer genomics. Cancer genomics & proteomics. 2018;15(1):41-51. PMID: 29275361
13. Hoerl RW. Ridge regression: a historical context. Technometrics. 2020;62(4):420-5. doi:10.1080/00401706.2020.1742207
14. Alhamzawi R, Ali HTM. The Bayesian elastic net regression. Communications in Statistics-Simulation and Computation. 2018;47(4):1168-78. doi: 10.1080/03610918.2017.1307399
15. Usman AU, Hassan S, Tukur K. Application of dummy variables in multiple regression analysis. International Journal of Recent Scientific Research. 2015;7(11):7440-2.
16. Berger VW, Zhou Y. Kolmogorov–smirnov test: Overview. Wiley statsref: Statistics reference online. 2014. doi: 10.1002/9781118445112.stat06558
17. González-Estrada E, Cosmes W. Shapiro–Wilk test for skew normal distributions based on data transformations. Journal of Statistical Computation and Simulation. 2019;89(17):3258-72. doi: 10.1080/00949655.2019.1658763
18. Saculinggan M, Balase EA, editors. Empirical power comparison of goodness of fit tests for normality in the presence of outliers. Journal of Physics: Conference Series; 2013: IOP Publishing. doi: 10.1088/1742-6596/435/1/012041
19. Patro S, Sahu KK. Normalization: A preprocessing stage. arXiv preprint arXiv:150306462. 2015. doi: 10.48550/arXiv.1503.06462
20. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research. 2002;16:321-57. doi: 10.1613/jair.953
21. Cao D, Chen Y, Chen J, Zhang H, Yuan Z. An improved algorithm for the maximal information coefficient and its application. Royal Society open science. 2021;8(2):201424. PMID: 33972855
22. Alhaj TA, Siraj MM, Zainal A, Elshoush HT, Elhaj F. Feature selection using information gain for improved structural-based alert correlation. PloS one. 2016;11(11):e0166017. doi: 10.1371/journal.pone.0166017
23. Akoglu H. User's guide to correlation coefficients. Turkish journal of emergency medicine. 2018;18(3):91-3. PMID: 30191186
24. Kornbrot D. Point biserial correlation. Wiley StatsRef: Statistics Reference Online. 2014. doi: 10.1002/9781118445112.stat06227
25. Lawrence S, Giles CL, Tsoi AC. What size neural network gives optimal generalization? Convergence properties of backpropagation: Citeseer; 1998.
26. Zollanvari A. Deep Learning with Keras-TensorFlow. Machine Learning with Python: Theory and Implementation: Springer; 2023. p. 351-91.
27. Belete DM, Huchaiah MD. Grid search in hyperparameter optimization of machine learning models for prediction of HIV/AIDS test results. International Journal of Computers and Applications. 2022;44(9):875-86. doi: 10.1080/1206212X.2021.1974663
28. Hile S, Erickson SJ, Agee B, Annett RD. Parental stress predicts functional outcome in pediatric cancer survivors. Psycho-Oncology. 2014;23(10):1157-64. PMID: 24817624
29. Bhatia S, Pappo AS, Acquazzino M, Allen-Rhoades WA, Barnett M, Borinstein SC, et al. Adolescent and Young Adult (AYA) Oncology, Version 2.2024, NCCN Clinical Practice Guidelines in Oncology. Journal of the National Comprehensive Cancer Network. 2023;21(8):851-80. PMID: 37549914
30. Leung KY, Lee HY. Implementing the smart city: Who has a say? Some insights from Hong

Kong. International Journal of Urban Sciences. 2023;27(sup1):124-48. doi: 10.1080/12265934.2021.1997634

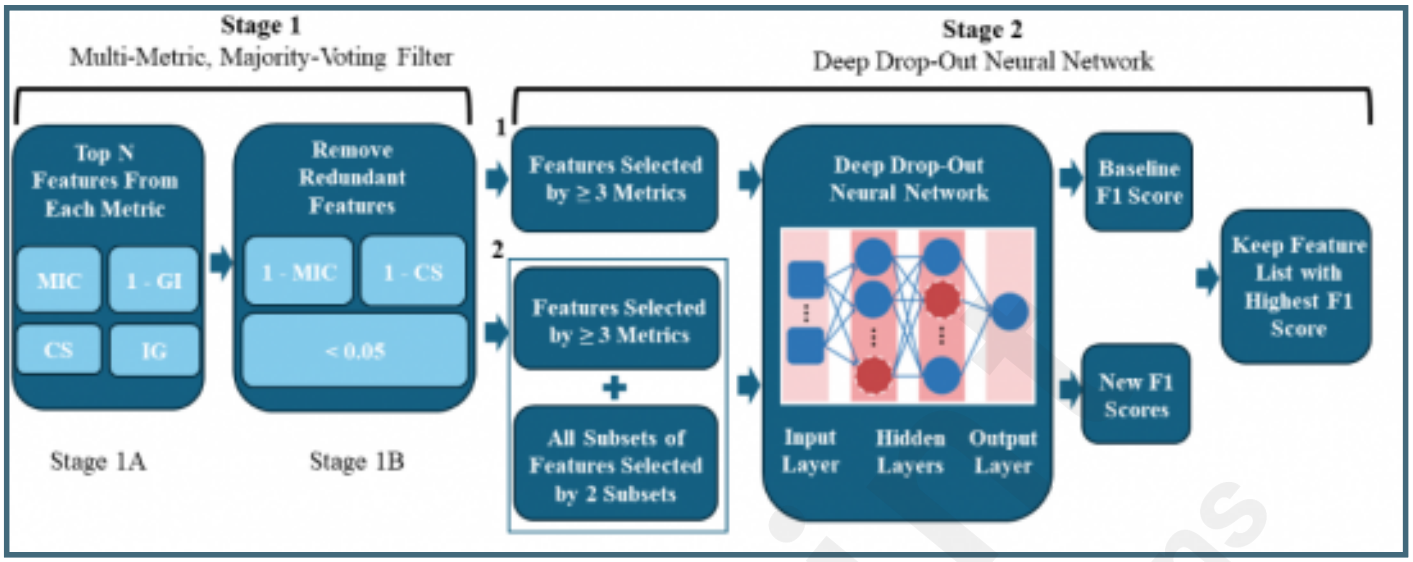
Supplementary Files

Figures

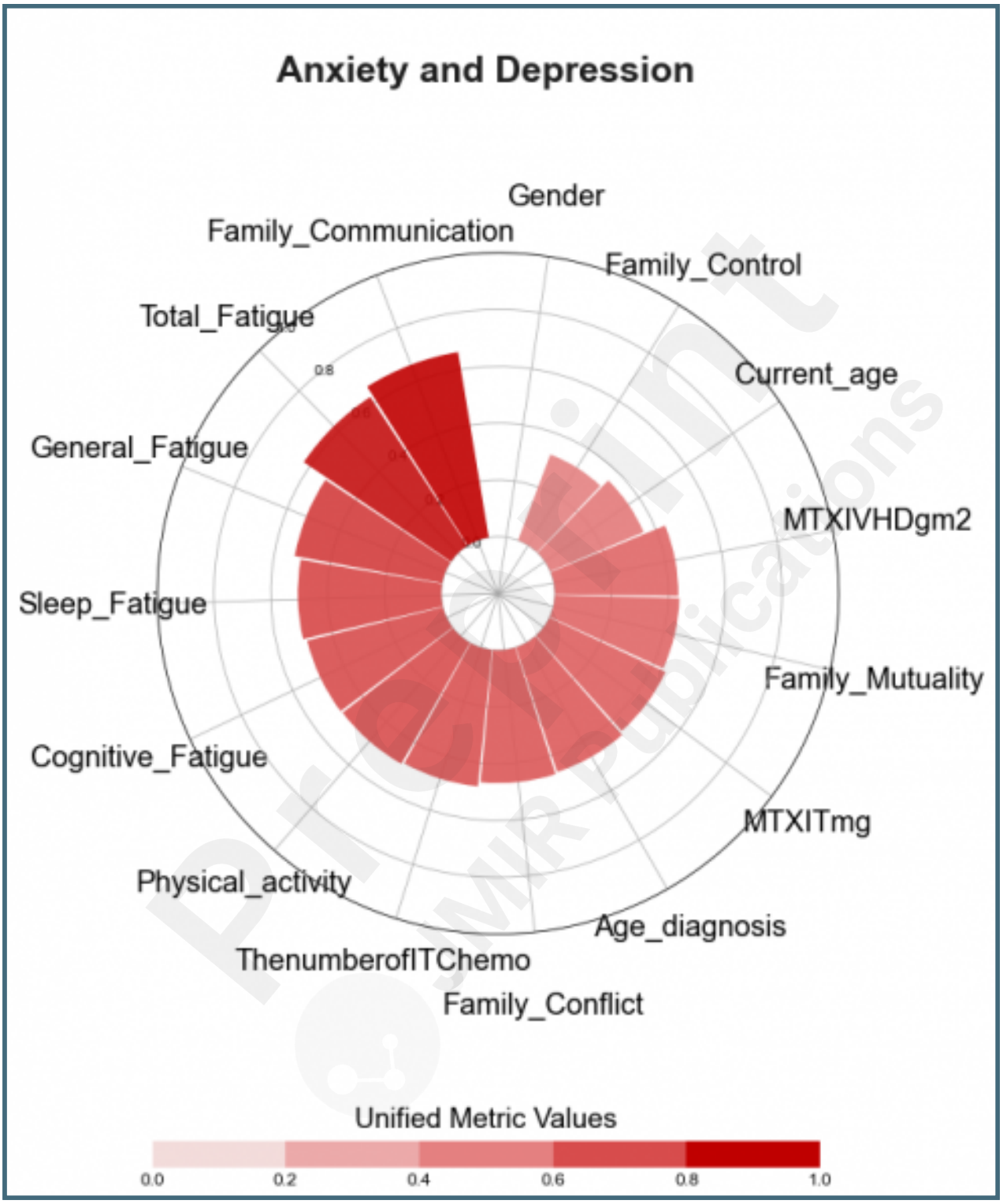
Data-driven, clinical domain-guided framework.



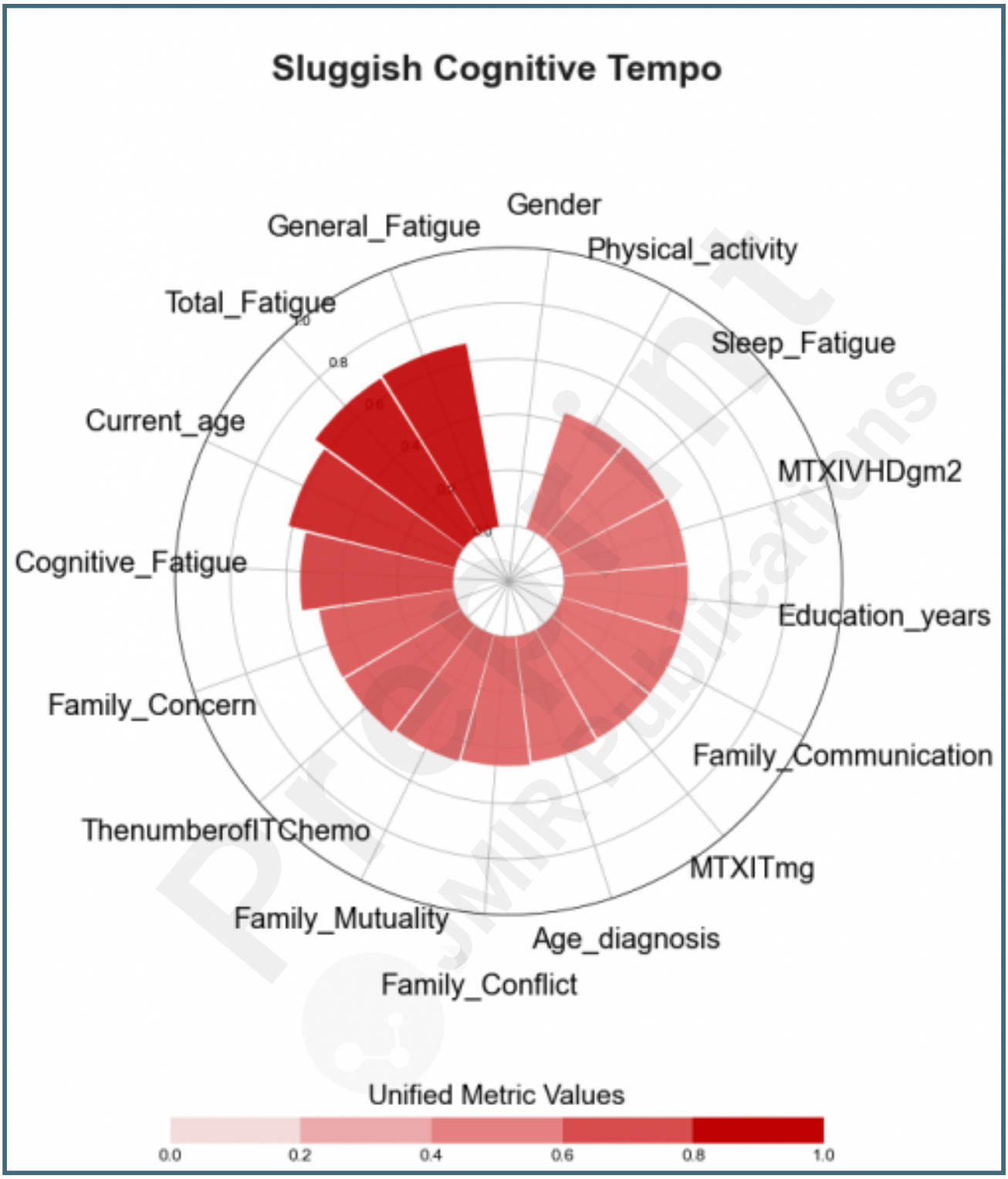
Two-stage feature selection algorithm.



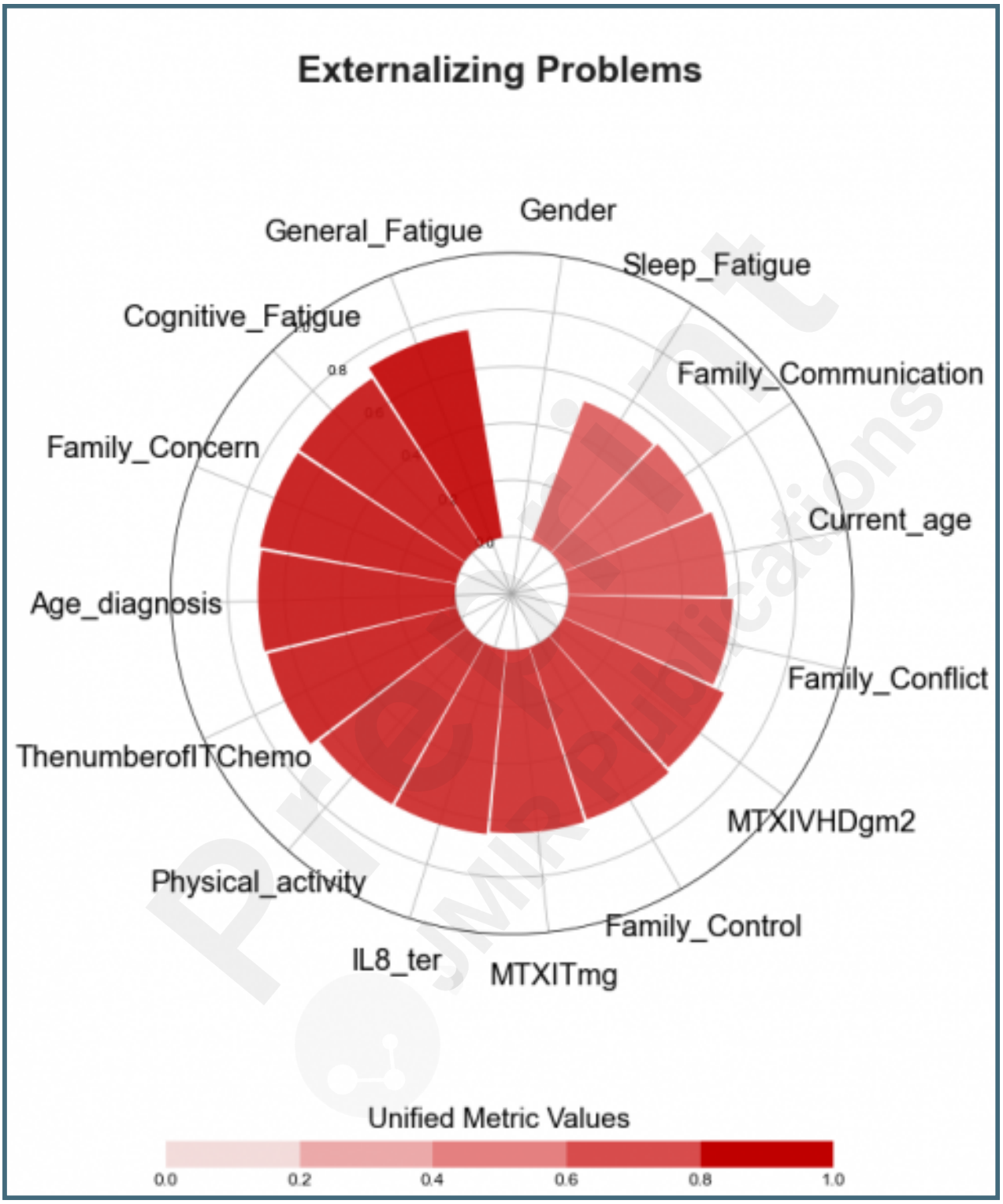
Anxiety and Depression Radial Feature Chart.



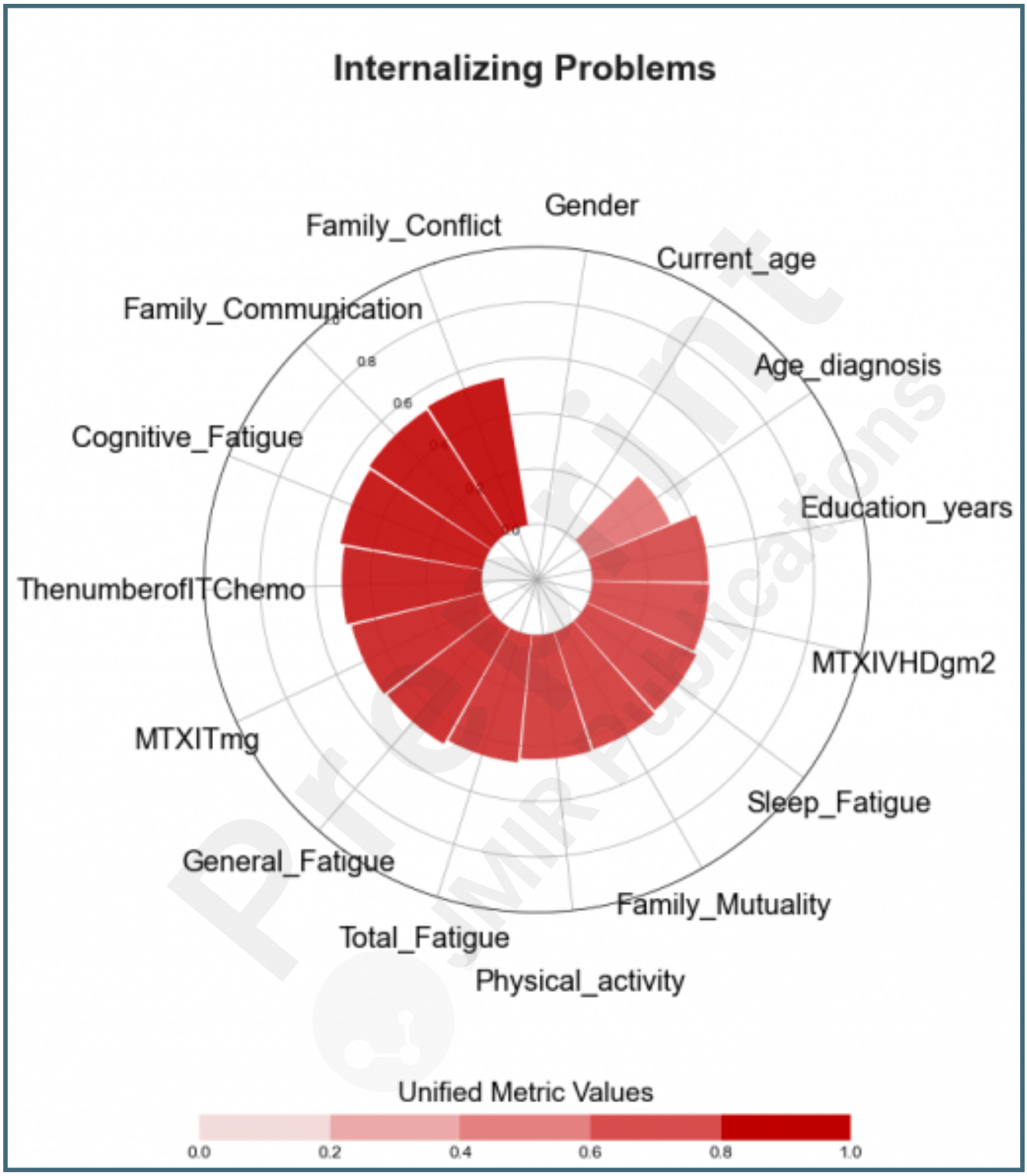
Sluggish Cognitive Tempo Radial Feature Chart.



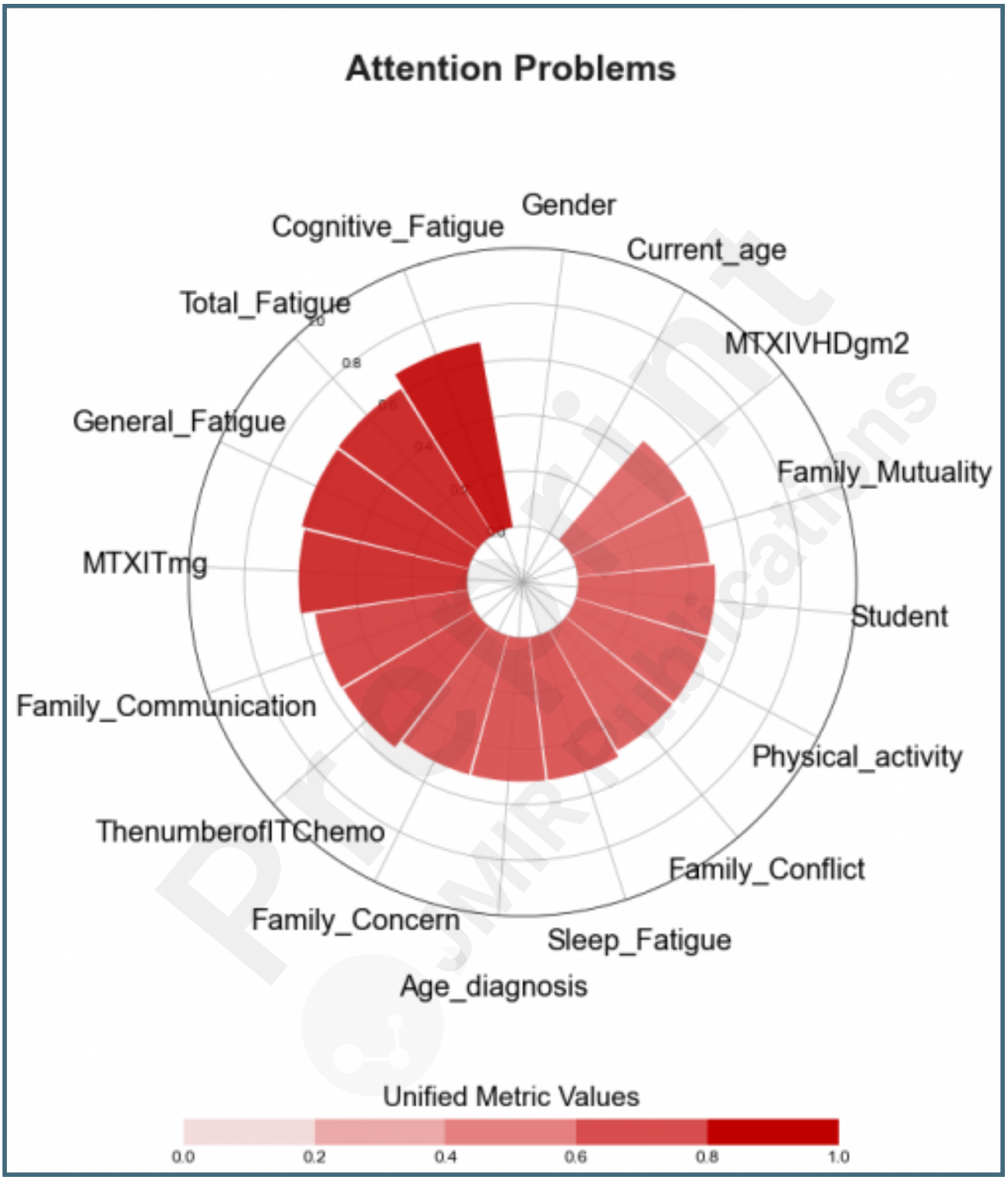
Externalizing Problems Radial Feature Chart.



Internalizing Problems Radial Feature Chart.



Attention Problems Radial Feature Chart.



Thought Problems Radial Feature Chart.