## **Final Group Project Report**

**Course:** Machine Learning - INT3405E\_56  
**Project Title:** Predicting Problematic Internet Usage from Children's Physical Activity Data  
**Team Name:** Nomad  
**Team Members:**

* Đào Thị Bình An - 22028326
* Bùi Thị Huyền Tâm - 22028214
* Đoàn Minh Quân - 20020152

## **1. Introduction**

### **1.1 Background**

This project is part of our coursework and involves participating in a Kaggle competition named “Child Mind Institute — Problematic Internet Use” to develop a predictive model that analyzes children’s physical activity data to predict levels of problematic internet usage. The dataset includes various physical fitness indicators that serve as the foundation for our machine learning model.

### **1.2 Motivation**

The project provides an opportunity to apply machine learning techniques to a real-world dataset, enhancing our understanding of the end-to-end process of data preprocessing, feature engineering, model building, and evaluation. By working on this challenge, we aim to gain practical experience and develop essential skills in handling machine learning projects.

### **1.3 Goals**

Our primary goal is to build a simple yet effective machine learning model that predicts problematic internet usage based on the provided dataset. Throughout the project, we focus on learning key concepts and improving our model iteratively to achieve better performance.

## **2. Dataset**

The dataset provided for this project consists of two main components:

**2.1. Tabular Data:**

* + Found in train.csv and test.csv, it includes demographic, physical, and behavioral measurements collected through various instruments. The fields within each instrument are described in data\_dictionary.csv. These instruments are:
    - **Demographics -** Information about age and sex of participants.
    - **Internet Use -** Number of hours of using computer/internet per day.
    - **Children's Global Assessment Scale -** Numeric scale used by mental health clinicians to rate the general functioning of youths under the age of 18.
    - **Physical Measures -** Collection of blood pressure, heart rate, height, weight and waist, and hip measurements.
    - **FitnessGram Vitals and Treadmill -** Measurements of cardiovascular fitness assessed using the NHANES treadmill protocol.
    - **FitnessGram Child -** Health related physical fitness assessment measuring five different parameters including aerobic capacity, muscular strength, muscular endurance, flexibility, and body composition.
    - **Bio-electric Impedance Analysis -** Measure of key body composition elements, including BMI, fat, muscle, and water content.
    - **Physical Activity Questionnaire -** Information about children's participation in vigorous activities over the last 7 days.
    - **Sleep Disturbance Scale -** Scale to categorize sleep disorders in children.
    - **Actigraphy -** Objective measure of ecological physical activity through a research-grade biotracker.
    - **Parent-Child Internet Addiction Test (PCIAT) -** 20-item scale that measures characteristics and behaviors associated with compulsive use of the Internet including compulsivity, escapism, and dependency; measuring internet addiction, with the derived target **sii** categorized as **None (0), Mild (1), Moderate (2), or Severe (3).**

**2.2. Actigraphy Data**:

* + Found in series\_train.parquet and series\_test.parquet, it contains accelerometer data collected from wrist-worn devices. Key features include:
    - **X, Y, Z**: Acceleration measurements along the three axes.
    - **enmo**: Euclidean Norm Minus One, indicating motion intensity.
    - **anglez**: Arm angle relative to the horizontal plane.
    - **light**: Ambient light in lux.
    - **non-wear\_flag**: Indicates whether the device was worn.
    - **Other Fields**: Time of day, day of the week, battery voltage, and relative date to the PCIAT test.

The target variable, **Severity Impairment Index (sii)**, is derived from the PCIAT score and represents the level of problematic internet usage.

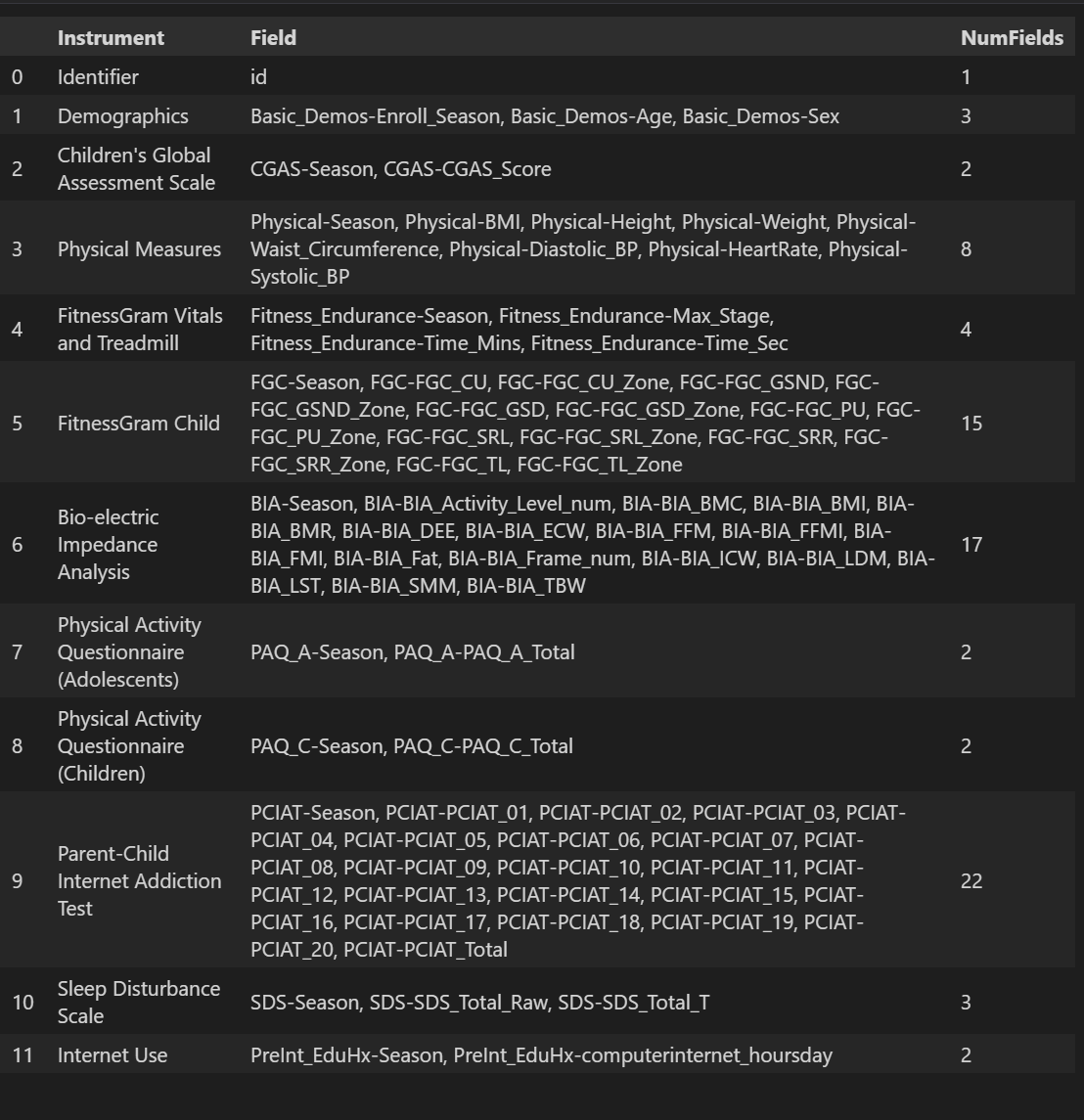
## **3. Methodology**

### **3.1. Exploratory Data Analysis (EDA)**

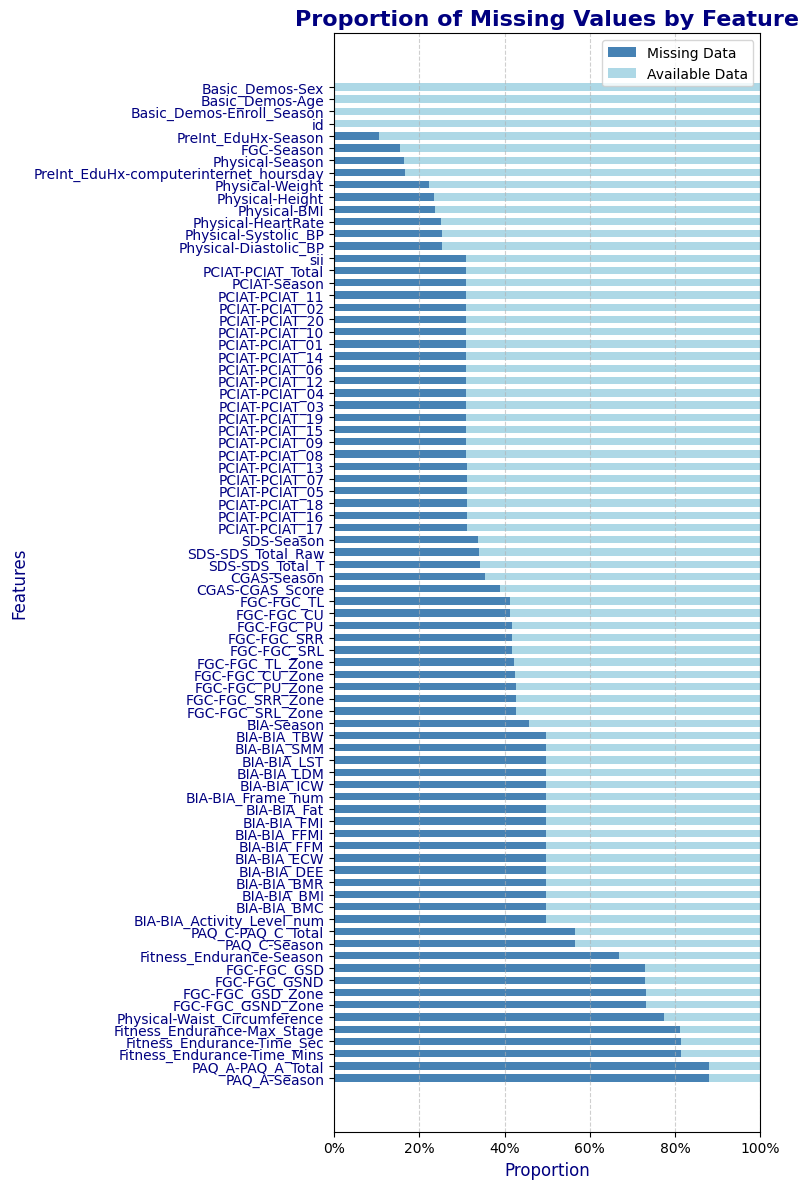
#### **3.1.1. Dataset Overview**

**Training Dataset:**

* + Contains 3960 samples and 80 features (excluding id and target sii). Features include demographic, physical activity, and fitness-related variables.



* Most features exhibit a high proportion of missing values, except for essential demographic attributes like id, sex, age, and season of enrollment.
* The target variable sii also has missing values (~30%) in the training set.



#### **3.1.2. Feature Classification & Distribution**

80 features is divided into 4 classes:

**Continuous: 24**

['Basic\_Demos-Age' 'Physical-BMI' 'Physical-Height' 'Physical-Weight'

'FGC-FGC\_GSND' 'FGC-FGC\_GSD' 'FGC-FGC\_SRL' 'FGC-FGC\_SRR' 'BIA-BIA\_BMC'

'BIA-BIA\_BMI' 'BIA-BIA\_BMR' 'BIA-BIA\_DEE' 'BIA-BIA\_ECW' 'BIA-BIA\_FFM'

'BIA-BIA\_FFMI' 'BIA-BIA\_FMI' 'BIA-BIA\_Fat' 'BIA-BIA\_ICW' 'BIA-BIA\_LDM'

'BIA-BIA\_LST' 'BIA-BIA\_SMM' 'BIA-BIA\_TBW' 'PAQ\_A-PAQ\_A\_Total'

'PAQ\_C-PAQ\_C\_Total']

**Discrete: 14**

['CGAS-CGAS\_Score' 'Physical-Waist\_Circumference' 'Physical-Diastolic\_BP'

'Physical-HeartRate' 'Physical-Systolic\_BP' 'Fitness\_Endurance-Max\_Stage'

'Fitness\_Endurance-Time\_Mins' 'Fitness\_Endurance-Time\_Sec' 'FGC-FGC\_CU'

'FGC-FGC\_PU' 'FGC-FGC\_TL' 'PCIAT-PCIAT\_Total' 'SDS-SDS\_Total\_Raw'

'SDS-SDS\_Total\_T']

**Categorical: 12**

['id' 'Basic\_Demos-Enroll\_Season' 'CGAS-Season' 'Physical-Season'

'Fitness\_Endurance-Season' 'FGC-Season' 'BIA-Season' 'PAQ\_A-Season'

'PAQ\_C-Season' 'PCIAT-Season' 'SDS-Season' 'PreInt\_EduHx-Season']

**Categorical Int: 31**

['Basic\_Demos-Sex' 'FGC-FGC\_CU\_Zone' 'FGC-FGC\_GSND\_Zone'

'FGC-FGC\_GSD\_Zone' 'FGC-FGC\_PU\_Zone' 'FGC-FGC\_SRL\_Zone'

'FGC-FGC\_SRR\_Zone' 'FGC-FGC\_TL\_Zone' 'BIA-BIA\_Activity\_Level\_num'

'BIA-BIA\_Frame\_num' 'PCIAT-PCIAT\_01' 'PCIAT-PCIAT\_02' 'PCIAT-PCIAT\_03'

'PCIAT-PCIAT\_04' 'PCIAT-PCIAT\_05' 'PCIAT-PCIAT\_06' 'PCIAT-PCIAT\_07'

'PCIAT-PCIAT\_08' 'PCIAT-PCIAT\_09' 'PCIAT-PCIAT\_10' 'PCIAT-PCIAT\_11'

'PCIAT-PCIAT\_12' 'PCIAT-PCIAT\_13' 'PCIAT-PCIAT\_14' 'PCIAT-PCIAT\_15'

'PCIAT-PCIAT\_16' 'PCIAT-PCIAT\_17' 'PCIAT-PCIAT\_18' 'PCIAT-PCIAT\_19'

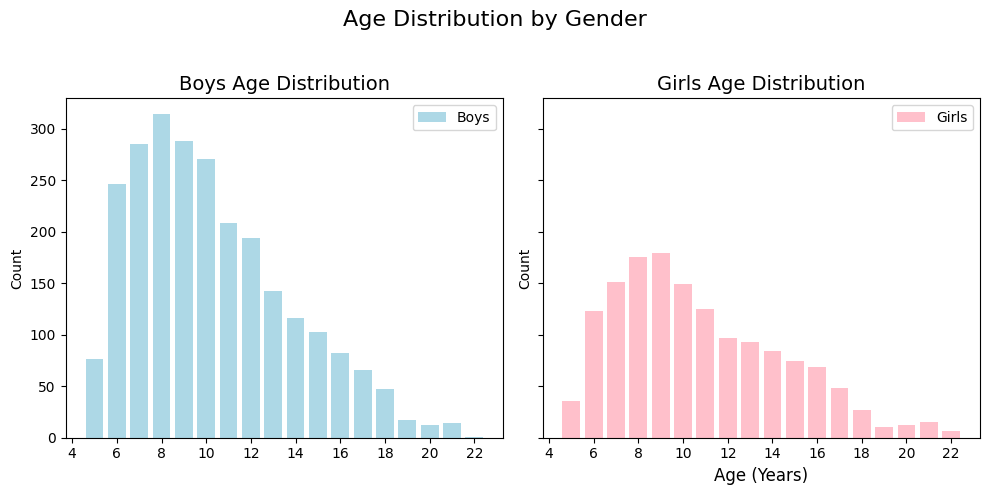
'PCIAT-PCIAT\_20' 'PreInt\_EduHx-computerinternet\_hoursday']

#### **Distribution of numerical features:**

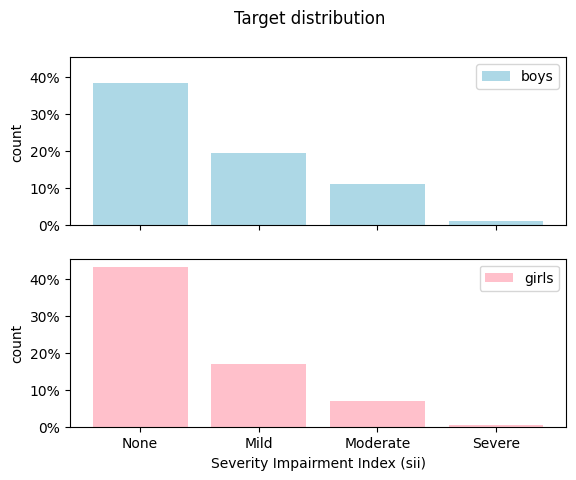
Look at some prominent information:

**a. Demographics**

* **Basic\_Demos-Age**:
  + Mean age is ~10.24 with a standard deviation of ~3.43, indicating that the dataset mainly consists of children and adolescents.
  + Age distribution ranges from 5 to 22, with the median (10) and interquartile range (8-12) showing most data is concentrated around school-age individuals. There are twice as many boys as girls.



* **Basic\_Demos-Sex**:
  + The mean value of ~0.36 (0 = male, 1 = female) suggests more male participants in the dataset and they have a slightly higher risk of internet addiction than girls.

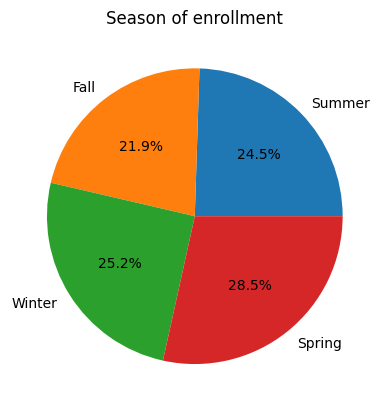


**b. Health and Physical Data**

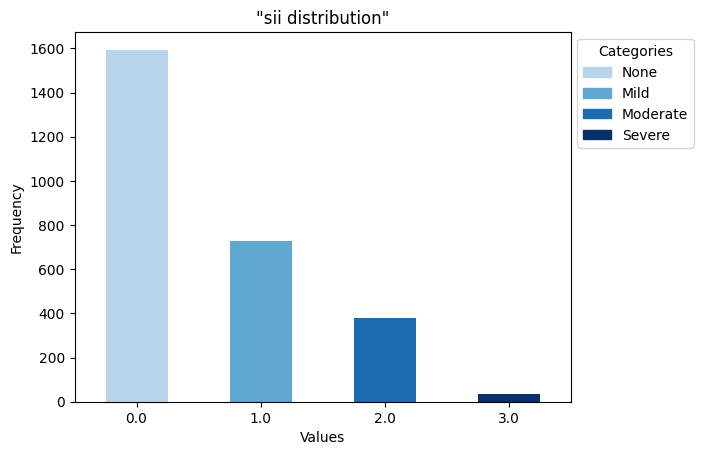
* **Physical-BMI**:
  + Mean BMI is ~19.1, typical for children and adolescents.
  + BMI of 0 is invalid and indicates null values in that sample.
* **Physical-HeartRate, Physical-Systolic\_BP, Physical-Diastolic\_BP**:
  + Average heart rate is ~81.82 bpm, and average systolic/diastolic blood pressure is ~117.12/69.75, which are within normal ranges.

#### **Distribution of season features:**

* Some columns such as **PAQ\_C-Season** and **PAQ\_A-Season** have very low observation, leading to unreliable seasonal distribution. **CGAS-Season** column also has a significant amount of data missing (2342/2736).
* Spring dominates in most columns, such as **PCIAT-Season, Basic\_Demos-Enroll\_Season, and Physical-Season.** This may reflect a seasonal bias in the data collection pattern. But Basic season of enrollment have similar frequencies.

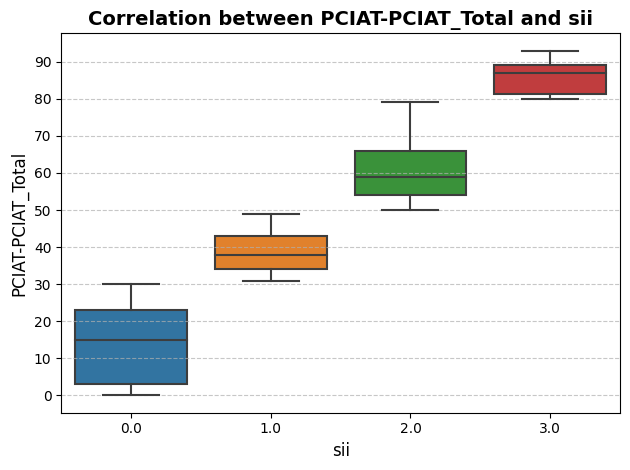


**Distribution of “sii”:**



* The target variable sii is imbalanced, with most participants falling into the "None" or "Mild" categories. If we use a regression approach, this does not matter much.

#### **3.1.3. Correlation Analysis**



The target sii is available exactly for those participants for whom we have results of the Parent-Child Internet Addiction Test (PCIAT), and it is a function of the PCIAT total score. That's why PCIAT results don't exist in test data.

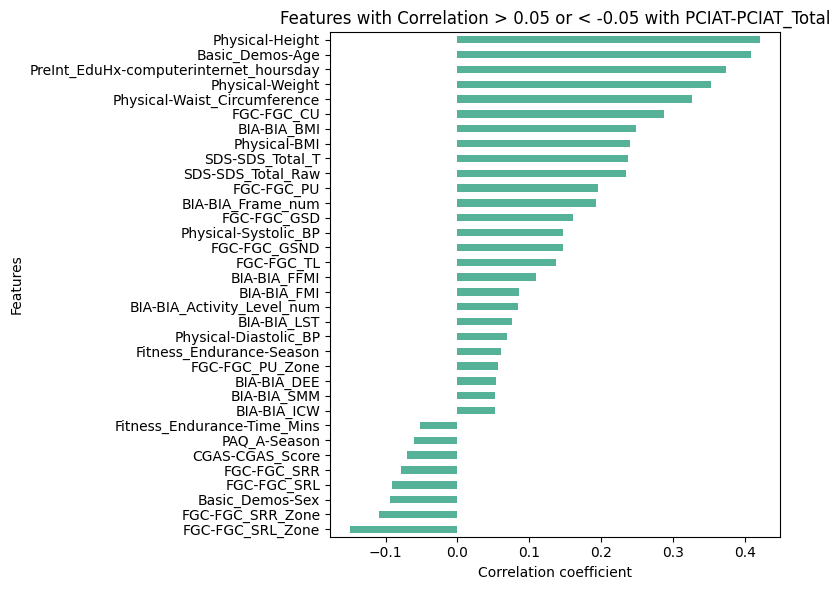
*- 0-30 Normal (0)*

*- 31-49 Mild (1)*

*- 50-79 Moderate (2)*

*- 80-100 Severe (3)*

* Therefore, we use correlation matrix to reveal relationships between numerical features and the target PCIAT Total scores (as sii).



**Insight:**

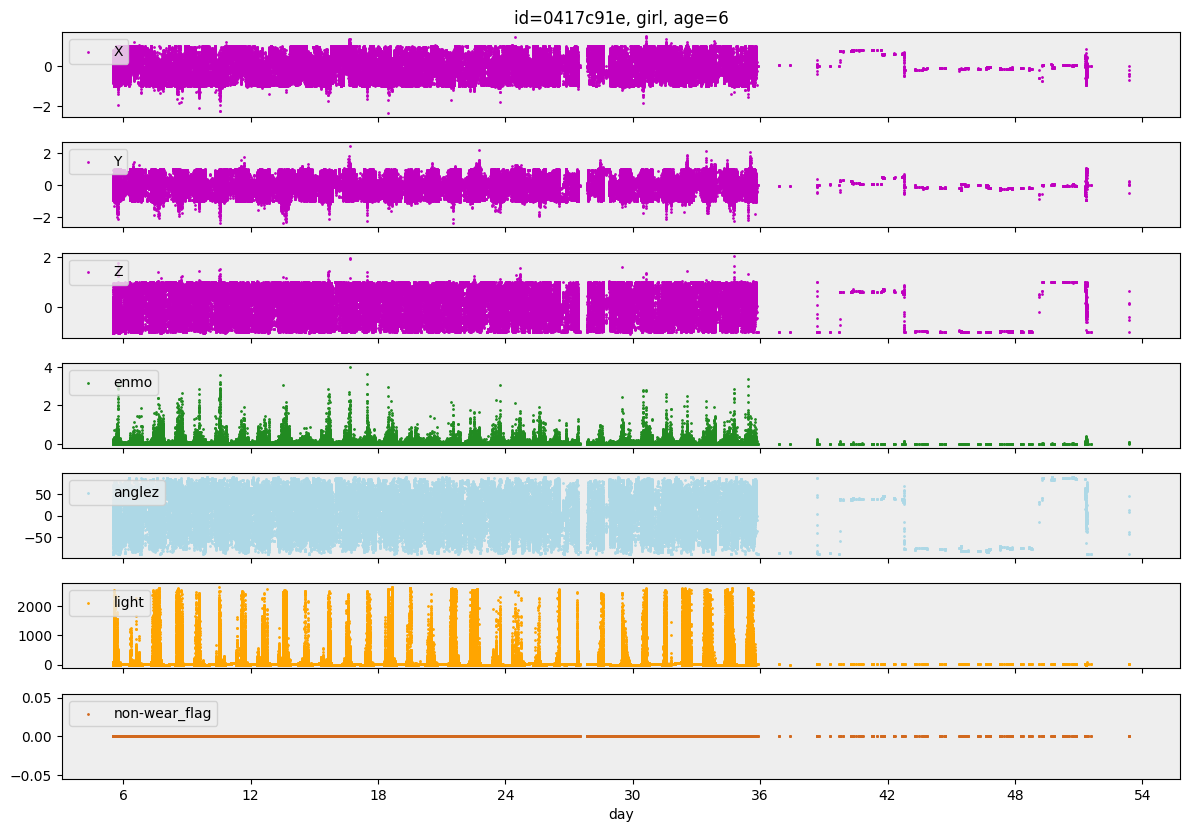
* We focus on predicting the target from all other features except the PCIAT results.
* We know the target only for two thirds of the samples. Therefore we use it to train supervised learning model.
* There are 2 approaches: directly predict sii or predict `PCIAT-PCIAT\_Total` and then transform this prediction to a sii prediction for submission.

#### **3.1.4. Actigraphy Data Insights**

Actigraphy is a non-invasive method of monitoring human rest/activity cycles. A small actigraph unit, also called an actimetry sensor, is worn for a week or more to measure gross motor activity. The unit is usually in a wristwatch-like package worn on the wrist. The movements the actigraph unit undergoes are continually recorded and some units also measure light exposure.

We have actigraphy files for a quarter of the participants (996 to be precise). The file name is always `part-0.parquet`.

Looking at the file of participant `id=0417c91e`, a six-year old right-handed girl, we see that this participant started to use the accelerometer on a Tuesday (weekday=2) of the second quarter of the year at second 44100 of the day (12:15 PM), 5 days after the PCIAT test. She gave the accelerometer back on the 53rd day after the PCIAT test, a Monday of the third quarter, at 9:08 AM. The competition data page says that `time\_of\_day` is in format `%H:%M:%S.%9f`. This is obviously not true. `time\_of\_day` is measured in nanoseconds since midnight.



We can plot diagrams of the time series in this file. We can see:

1. We clearly see a daily pattern.

2. We see that the girl wore the accelerometer for 31 days and then took it off.

3. The dataset has a non-wear\_flag column, but that flag is always zero for this participant.

4. The girl is in an environment where the illuminance exceeds 2500 [lux](https://en.wikipedia.org/wiki/Lux) every day (the device cannot measure more than 2500 lux). Such a high illuminance means that she is outdoors or in a room with huge windows.

5. The girl moves a lot: she has enmo values above 2 almost every day.

6. The time series usually contain measurements every 5 seconds, but some time steps are missing. It is not documented under what conditions time steps are skipped.

**Insight:**

- Some columns are not really useful.

- The ENMO and light columns offer themselves for analysis with a one-dimensional convolutional neural network, but if we want to start simple, we can use some basic aggregations (mean, variance, ...) of the time series as features for a gradient-boosting model.

### **3.2. Preprocessing & Feature engineering**

#### **3.2.1**. **Handling Missing Data in the "sii" Column**

Initially, there were 2736 non-null values in the sii column, and missing values represented approximately 30% of the data.

Given the high proportion of missing data in this column, the decision was made to drop the records with missing sii values. This allowed for accurate training of the supervised model using only valid data.

The following code was used to drop records with missing sii values:

usable\_train\_df = train\_df.dropna(subset='sii')

#### **3.2.2**. **Encoding Seasonal Columns**

The columns related to seasons: *'Basic\_Demos-Enroll\_Season', 'CGAS-Season', 'Physical-Season', 'Fitness\_Endurance-Season', 'FGC-Season', 'BIA-Season', 'PAQ\_A-Season', 'PAQ\_C-Season', 'PCIAT-Season', 'SDS-Season', 'PreInt\_EduHx-Season'*

were encoded using Label Encoding to convert categorical season labels into numerical values.

For each column in the list columns\_to\_encode, we filter non-null values, convert them ​​into string, and then use LabelEncoder to encode the unique values ​​from training data. If the value is empty (NaN), keep it NaN.

Fall → 0

Spring → 1

Summer → 2

Winter → 3

We applied LabelEncoder on both training and testing data frames.

#### **3.2.3**. **Processing Actigraphy Data**

##### **3.2.3.1. Loading data**

We construct the **load\_time\_series** function to load and process multiple time-series data files in parallel from a specified directory. It uses the ThreadPoolExecutor to speed up the process by processing files concurrently, based on the number of available CPU cores. Each file is processed by the **process\_file** function. The **process\_file** function was used to read and clean each file by removing unnecessary columns (**step** column), and the results were aggregated into a final DataFrame with descriptive statistics (like mean, variance, max, min, std,...). The function returns the DataFrame containing these processed features and file identifiers.

##### **3.2.3.2**. **Encode for Actigraphy Data**

We applied an **AutoEncoder**, which is a type of neural network used for unsupervised learning and dimensionality reduction to process the actigraphy data. Since actigraphy data has many features, not all of which are useful, the AutoEncoder reduces the data’s dimensionality by encoding it into a smaller set of features. It also learns to discard noise while retaining important patterns.

An **AutoEncoder** consists of two main components:

* **Encoder**: Compresses the input data into a lower-dimensional representation using 3 layers with ReLU activation, which helps the network learn complex patterns.
* **Decoder**: Reconstructs the original input data from the compressed representation. It uses the reverse order of layers, with ReLU applied in the first two layers and a Sigmoid activation function in the output layer.

The autoencode\_data function trains the AutoEncoder. First, the data is scaled using **StandardScaler** to ensure all features are on the same scale. Then, the AutoEncoder compresses the data into a 50-dimensional representation. The model is trained with the **Adam optimizer** and **Mean Squared Error (MSE)** loss function. After training, the encoded features (Acti) are added to a new DataFrame for later use.

#### **3.2.4**. **Feature Selection**

We selected 58 features from the testing DataFrame. To ensure the quality of the data, we filtered out any columns with missing values greater than 50%. Finally, we merged the selected features with the DataFrame containing the actigraphy data on common **id** values., combining them into a comprehensive dataset for training called **selected\_df with shape (2736,97).**

#### **3.2.5**. **Imputation of Missing Data**

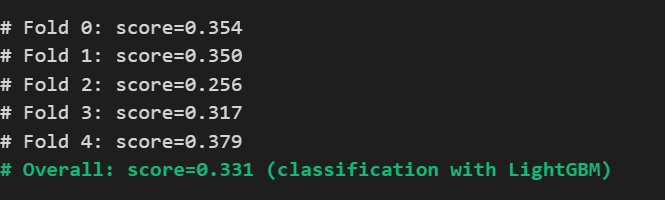
Missing data in the final dataset was handled by **IterativeImputer**, ensuring that the model could train on a complete dataset without losing valuable information. In **Iterative Imputation**, each feature with missing values is predicted by modeling it as a function of other features in the dataset. The missing values are replaced in each iteration, and the model updates these replacements.

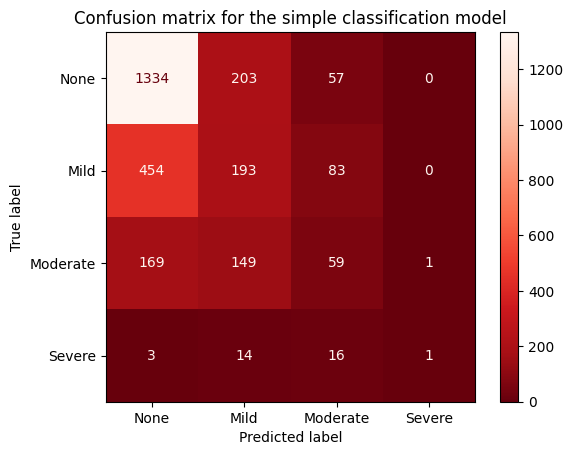
### **3.3. Model Selection**

In this study, we initially tested both classification and regression models, even though the problem is a classification task. With a regression model, even though it produces continuous outputs, we can later be rounded to discrete values.

* **Classification Model**: We used a **LightGBM Classifier** for the initial test. The model was evaluated using **Cohen's Kappa score**, training through 5 folds by using StratifiedKFold.

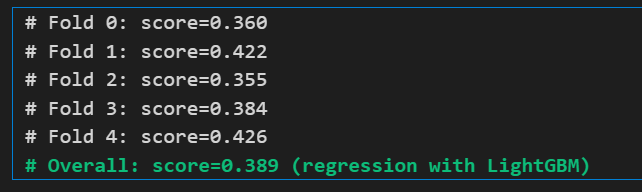
Here are the result:

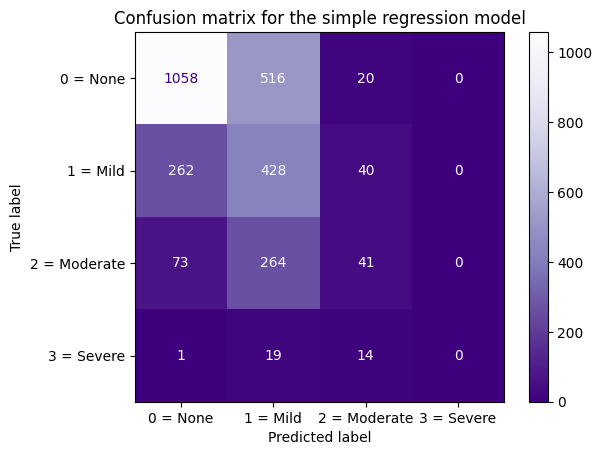




* **Regression Model**: A **LightGBM Regressor** was then employed. The continuous output of the regression model was rounded to integer values to match the expected format for the task. The model also was evaluated using Cohen's Kappa score, training through 5 folds by using StratifiedKFold.

Here are results:



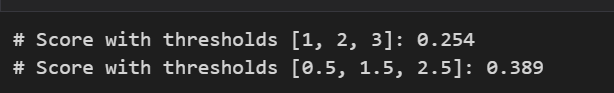


* The Kappa score for the regression model was 0.389, which was higher than the classification model, indicating that the regression model performed better for this particular task. The confusion matrix shows that our regression model never predicts the class 'Severe' (the rightmost column of the matrix is zero).

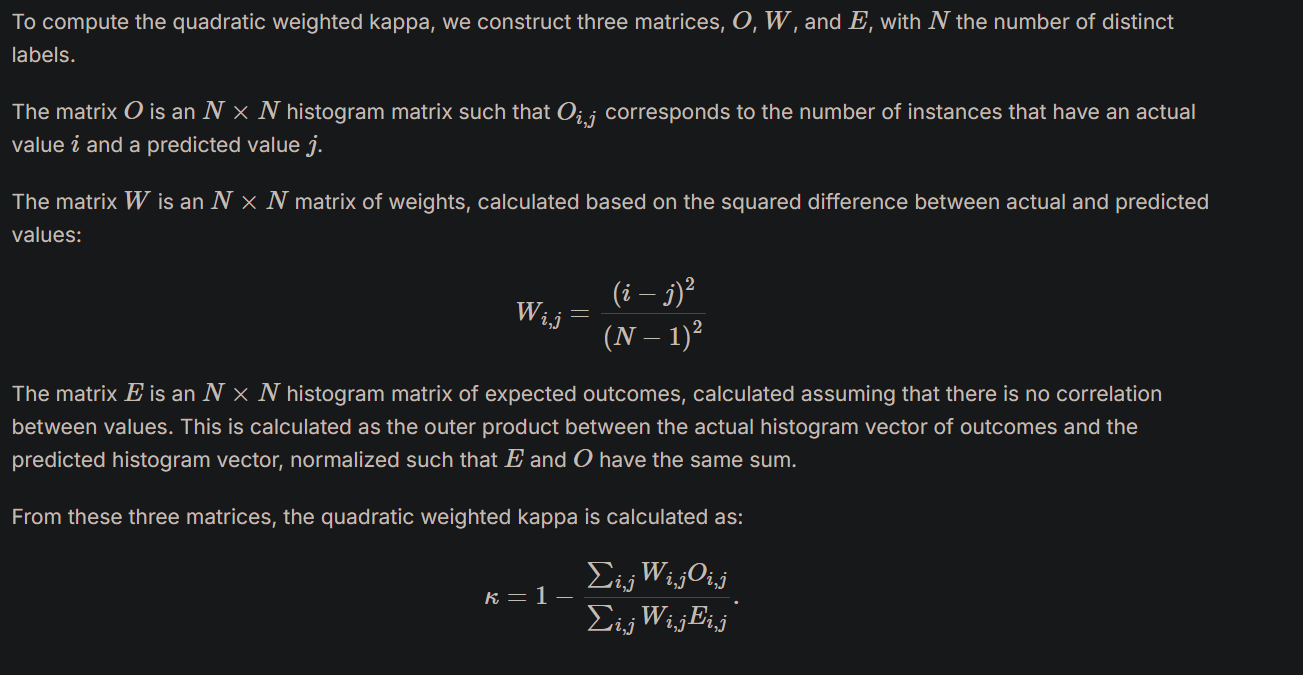
=> **We choose to build a regression model.**

So, our regression model predicts float values, and in the previous section of this notebook, we rounded these float values to integers because the Kaggle competition expects integer predictions.

Use the thresholds of 0.5, 1.5, and 2.5 is not necessarily the best approach for achieving the highest score. We could use different thresholds, like 1, 2, and 3, but in practice, the thresholds of 0.5, 1.5, and 2.5 actually yield a significantly higher score.

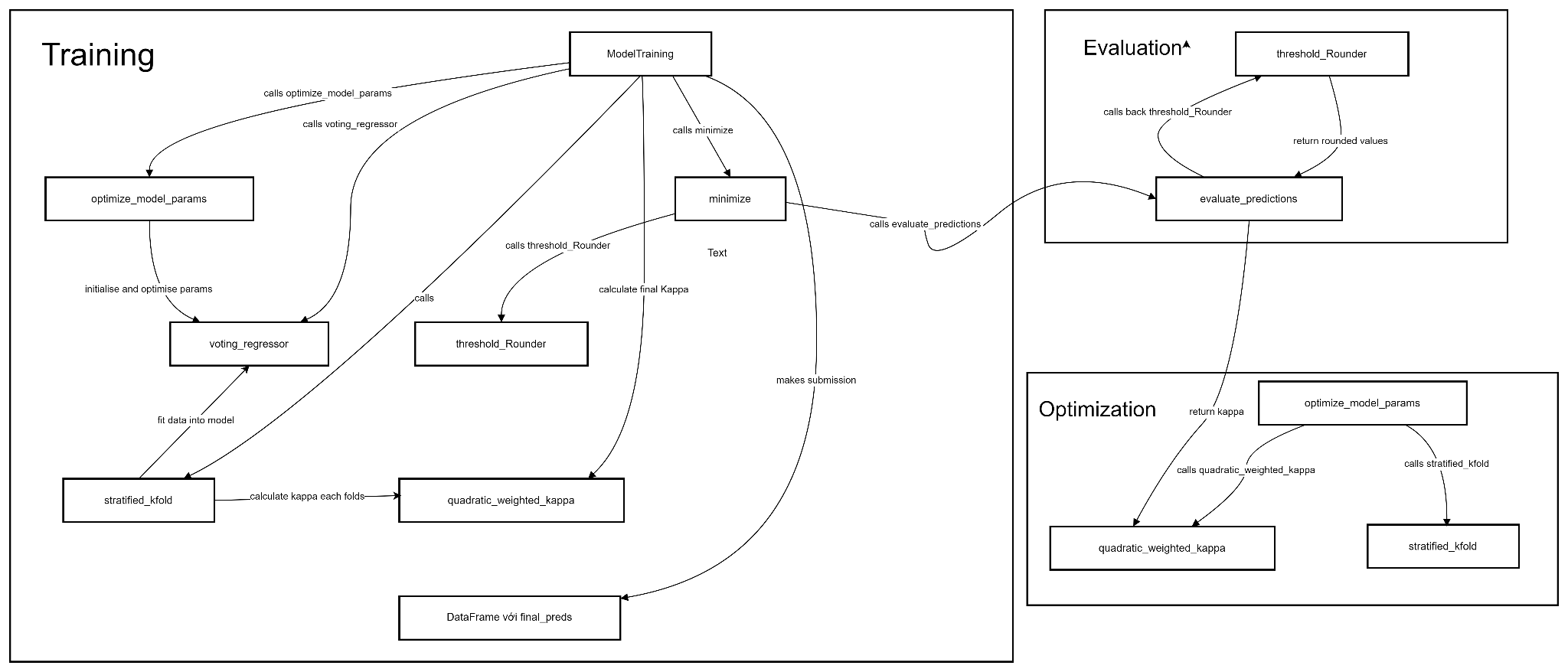
****

### **3.4. Evaluation Metrics**

We evaluated the performance of our models using Cohen’s Kappa score, which measures the agreement between predicted and actual values while accounting for class imbalances. Specifically, we used the **quadratic weighted Kappa** to penalize larger errors more heavily. Moreover, we used **Stratified K-fold cross-validation**, ensuring that the performance metrics were robust and reliable.

## **4. Model implementation**

In this section, we describe the implementation of the machine learning model, from parameter optimization to training and prediction. The model used for this task is an ensemble of regression models, optimized using cross-validation and hyperparameter tuning.



### **4.1. Model Selection and Hyperparameter Optimization**

First, we used three different regression models—**LightGBM**, **XGBoost**, and **CatBoost**—as base learners in a **Voting Regressor** ensemble. Before training the models, we optimized their hyperparameters using **Optuna**, a hyperparameter optimization library. The parameters for each model were chosen to maximize performance based on the **quadratic weighted Kappa (QWK)** score, which is particularly useful for ordinal classification tasks like this one.

**Optimization Function**: The optimize\_model\_params function was used to perform hyperparameter optimization. This function defines the range of hyperparameters for each model (LightGBM, XGBoost, and CatBoost), and then uses **Stratified K-fold cross-validation** to evaluate the model performance. The goal is to minimize the negative mean QWK score, with the best parameters being selected after multiple trials.

### **4.2. Model Training**

The function **ModelTraining** takes three inputs:

* X: The feature set (independent variables) for training.
* y: The target variable (dependent variable) for training.
* test\_data: Data used for making predictions after the model is trained.

=> It returns a DataFrame containing predictions for the test data.

The steps in the **ModelTraining** function:

* Firstly, optimize parameters for each model are obtained from the **optimize\_model\_params** function, and the models are initialized with these parameters.
* After training 3 base models, they are combined into a **Voting Regressor**. This ensemble model predicts continuous values, and the **Voting Regressor** helps improve accuracy by aggregating the predictions from the individual models.
* To train the model, we use **Stratified K-Fold cross-validation**, which splits the data into 10 parts while ensuring that each fold has a similar distribution of the target variable. For each fold, the training and validation sets are created, and the **VotingRegressor** is trained on the training data. The predictions on the validation data are stored in **oof\_non\_rounded** for out-of-fold evaluation, while predictions on the test data are stored in **test\_preds**.
* After training the model, the continuous predictions are converted into discrete values using a thresholding function. The **threshold\_Rounder** function is used to round the predictions based on predefined thresholds, which are optimized using the **Nelder-Mead optimization method**. The thresholds are optimized to maximize the Kappa score by minimizing the error between predicted and actual values.

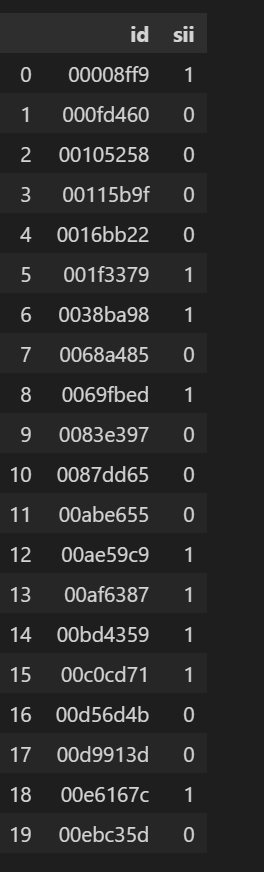
### **4.3. Final Predictions and Submission**

Once the optimal thresholds are determined, the final predictions are made on the test dataset. The test predictions are aggregated and averaged across all folds. The predictions are then rounded using the optimized thresholds, and the final results are saved in the format required for submission.

## **5. Results**

### **5.1. Prediction results**

Our result : 11 sii 0 and 9 sii 1 on test.csv

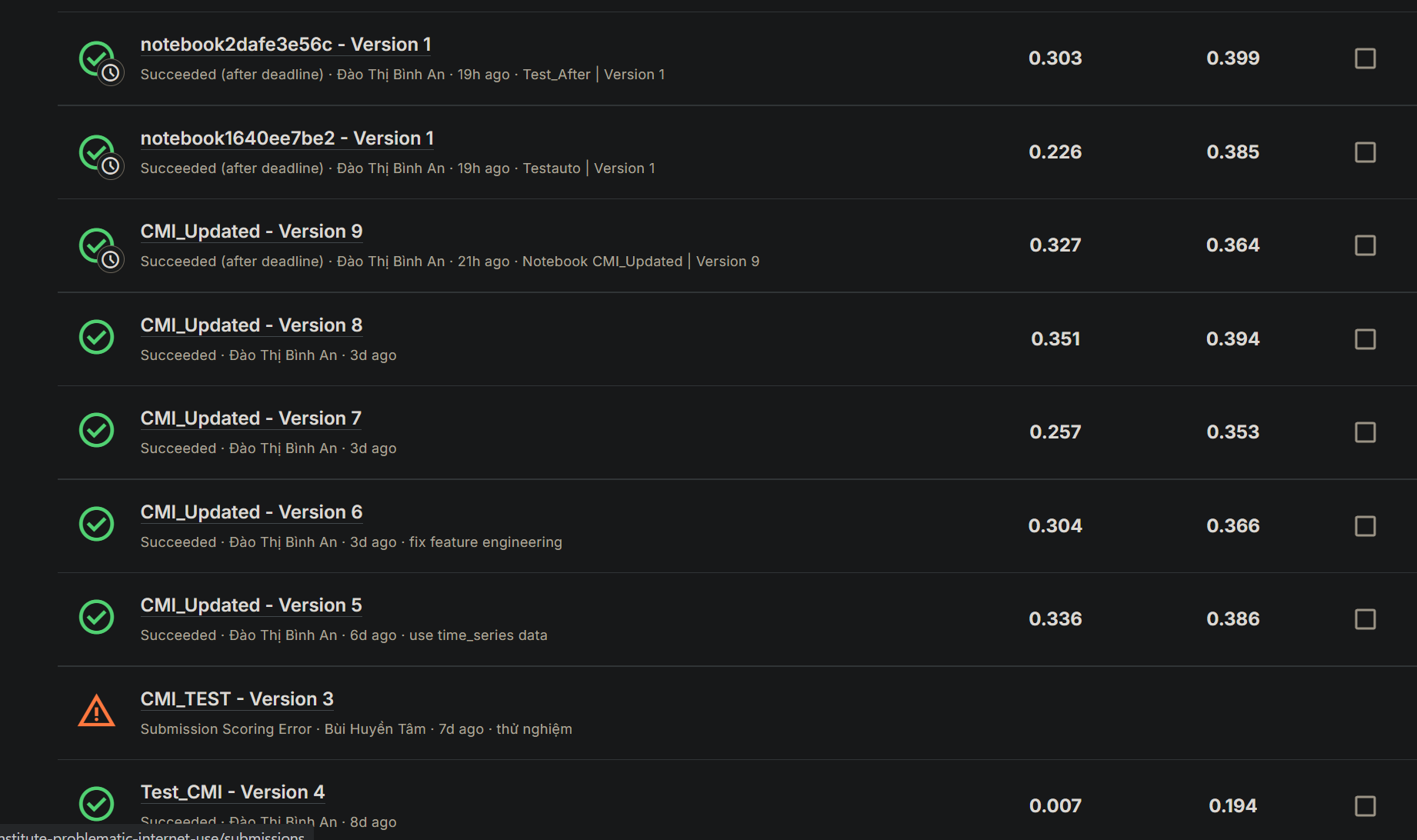


**=>** Public score on Kaggle is: **0.399**

### **5.2. Comparison**

This is a comparison of some of the progressing versions we have submitted to the competition to see the improvement of the new model compared to the old models.

| Model version | Kaggle Public Score | Runtime | Key improvement |
| --- | --- | --- | --- |
| Test\_CMI Version 4 | 0.194 | 72.4s | * It is the first model, actigraphy data was not used. * Features chosen for selection were only 16 columns that have a correlation with Total\_PCIAT: *SDS-SDS\_Total\_Raw, BIA-BIA\_BMI, FGC-FGC\_TL, BIA-BIA\_Frame\_num, Physical-BMI, SDS-SDS\_Total\_T, BIA-BIA\_FFMI, Physical-Height, FGC-FGC\_SRL\_Zone, PreInt\_EduHx-computerinternet\_hoursday, Basic\_Demos-Age, Physical-Systolic\_BP, FGC-FGC\_CU, FGC-FGC\_SRR\_Zone, Physical-Weight, FGC-FGC\_PU.* * The encoding of the season data failed (NaN values were not handled, so NaNs were set to 4 by default, which corresponds to the fifth season value). |
| CMI\_Updated Version 5 | 0.386 | 176.7s | * The issue with encoding the season data has been resolved. * Actigraphy data was used and encoded using an AutoEncoder. * The number of features selected for training (excluding Acti) is 29:   *Basic\_Demos-Age, Physical-Systolic\_BP, BIA-BIA\_FFMI, BIA-BIA\_BMI, FGC-FGC\_SRR, Physical-Weight, Physical-Diastolic\_BP, BIA-BIA\_ICW, BIA-BIA\_LST, SDS-SDS\_Total\_Raw, Physical-BMI, SDS-SDS\_Total\_T, FGC-FGC\_SRR\_Zone, FGC-FGC\_PU\_Zone, FGC-FGC\_TL, CGAS-CGAS\_Score, Basic\_Demos-Sex, FGC-FGC\_SRL, FGC-FGC\_SRL\_Zone, PAQ\_C-Season, BIA-BIA\_Activity\_Level\_num, Physical-Height, BIA-BIA\_Frame\_num, PreInt\_EduHx-computerinternet\_hoursday, BIA-BIA\_SMM, BIA-BIA\_DEE, BIA-BIA\_FMI, FGC-FGC\_PU, FGC-FGC\_CU"* |
| CMI\_Updated Version 8 | 0.394 | 234.3s | * All the features in Test data were chosen (58) then filtered with missing (remaining 48). |
| CMI\_final | 0.399 | 362.0s | * Before training, construct a function to optimize model hyperparameters using Optuna to maximize the quadratic weighted Kappa (QWK) score. The optimize\_model\_params function defined parameter ranges for LightGBM, XGBoost, and CatBoost, using Stratified K-fold cross-validation to select the best parameters by minimizing the negative mean QWK score. |



## **6. Conclusion and Future Work**

### **6.1. Summary**

This project successfully developed a predictive model for assessing problematic internet usage by utilizing a robust dataset and advanced machine learning techniques. Key contributions include the integration of Actigraphy data through AutoEncoders, hyperparameter optimization using Optuna, and the implementation of a Voting Regressor combining LightGBM, XGBoost, and CatBoost. These methods significantly enhanced the quadratic weighted Kappa (QWK) score, demonstrating the model's ability to predict outcomes with high accuracy.

### **6.2. Challenges**

During the development process, several challenges were encountered:

1. **Data Quality Issues:** Handling missing values and encoding errors, required additional preprocessing efforts.
2. **Feature Selection:** Selecting relevant features while excluding redundant or highly correlated ones posed difficulties due to the complexity of the dataset.
3. **Hyperparameter Optimization:** Balancing computational efficiency with achieving optimal performance during model tuning was resource-intensive.
4. **Actigraphy Integration:** Successfully encoding Actigraphy data using AutoEncoders presented technical challenges due to the high dimensionality and sparsity of the data.

### **6.3. Future Work**

Several improvements and extensions can be made to enhance the model further:

1. **Feature Analysis:** Perform a more detailed analysis of feature correlations to refine the selection process and better understand their relationships with the target variable.
2. **Outlier Handling:** Develop more sophisticated techniques to detect and remove outliers, ensuring the robustness of the model.
3. **Feature Engineering:** Create new features derived from existing ones to capture additional patterns and enhance predictive power.
4. **Model Variants:** Explore advanced ensemble methods and neural network architectures for comparison and potential performance gains.
5. **Real-World Testing:** Validate the model using external datasets or real-world data to assess its generalizability and practical utility.

## **7. References**

[1] [https://www.kaggle.com/competitions/child-mind-institute-problematic-internet-use](https://www.kaggle.com/competitions/child-mind-institute-problematic-internet-use/submissions)/

[2] <https://catboost.ai/docs/en/installation/python-installation-method-pip-install>

[3] <https://www.datacamp.com/tutorial/introduction-to-autoencoders>

[4] <https://digitalwellnesslab.org/wp-content/uploads/Scoring-Overview.pdf>