Machine Learning in Satellite Imagery and Other Geotagged Data Sources for Health Monitoring in Low- and Middle-Income Countries

Summer Research Internship, 2023
Department of Primary Care and Population Health
Stanford University, School of Medicine

Vedant Zope
Indian Institute of Technology Kharagpur
vedantzope@kgpian.iitkgp.ac.in

Soham Tripathy
Indian Institute of Technology Kharagpur
soham@kgpian.iitkgp.ac.in

Kushagra Parmeshwar
Indian Institute of Technology Kharagpur
kushgra.parmeshwar@kgpian.iitkgp.ac.in

Abstract

This research project pioneers the integration of machine learning and satellite imagery to monitor health indicators in low- and middle-income countries (LMICs). Utilizing a comprehensive dataset from the Google Earth Engine and Worldbank API, we developed a model that predicts critical health indicators pinpointed accurately to latitude and longitude. Our innovative approach addresses the data gaps prevalent in traditional health surveys, employing advanced machine learning regression models to expedite computations on standard hardware. The project has achieved an MCRMSE score of 10.9173, demonstrating the potential of this method for health monitoring in LMICs. This work contributes significantly to public health, showcasing the potential of satellite imagery and machine learning in delivering valuable insights into health conditions within resource-constrained settings.

1 Introduction

The fusion of satellite imagery and machine learning techniques has emerged as a potent instrument for monitoring and interpreting various global phenomena. Its application in the realm of public health, particularly, holds the potential to transform the surveillance and evaluation of health indicators in low- and middle-income countries (LMICs). This project is at the forefront of this transformation, harnessing the capabilities of satellite imagery and machine learning to monitor and predict key health indicators in LMICs. The ultimate objective is to equip policymakers and organizations with the necessary insights to effectively make informed decisions and allocate resources.

Monitoring health indicators in LMICs presents a myriad of challenges. The limited access to reliable, up-to-date health data, compounded by resource constraints, hinders the acquisition of comprehensive insights into the health status of these populations. Traditional data collection and analysis methods, which rely heavily on surveys and manual data entry, are not only time-consuming but also prone to data gaps and inaccuracies. Therefore, the need for innovative, efficient, and accurate approaches to health indicator information in LMICs is more critical than ever.

In response to this need, our project leverages a rich dataset, `gee-features.csv`, derived from Demographic and Health Surveys (DHS) conducted in 59 countries over a decade. This dataset is enriched by incorporating satellite imagery data from the Google Earth Engine API. To improve our model's performance, we have augmented our data using external sources like the World Bank. The data was queried using the `wbdata API` (Python module) by feeding in the country and the year, and then finally integrated it with the initial data.

Our approach primarily focuses on machine learning regression models to forecast critical health indicators, including *Mean BMI, Median BMI, Unmet Need Rate, Under-5 Mortality Rate, Skilled Birth Attendant Rate, and Stunted Rate* . Instead of deep learning methods which are computationally expensive and require large amounts of data, we have focused our work on using ensemble regression methods. These models expedite computations and facilitate predictions on standard computers, striking an optimal balance between prediction accuracy and computational efficiency. This equilibrium ensures the accessibility and feasibility of our models for deployment in LMICs.

Through data preprocessing, feature engineering, feature transformation methods, and a myriad of imputation techniques, we have achieved a remarkable MCRMSE of 10.9173 on the Kaggle leaderboard. Fine-tuning hyperparameters using Optuna and running various experiments to extract the optimal parameter values aided in mitigating the error.

This report offers a detailed analysis of our methodology, dataset, experimental setup, and results. We also discuss the limitations, challenges, and opportunities for future research in this field. By illuminating the potential of satellite imagery and machine learning in the context of public health, this project contributes to the burgeoning body of knowledge in data-driven approaches for health monitoring and resource allocation.

In the subsequent sections, we provide an extensive overview of related work, describe our methodology, present the dataset, discuss the experimental setup, and analyze the results. The report concludes with a summary of our findings, implications for practice, and future research directions.

2 Related Work

Our work builds on a growing body of research that seeks to automatically extract health indicators from satellite imagery data. Most existing approaches involve feature extraction from Convolution Neural Networks (CNNs) and predicting health indicators using Machine Learning or Deep Learning algorithms. For example, Piaggesi et al. estimated poverty using satellite data after feature extraction using CNNs [10]. Mastelini et al. used regressor chaining to enhance the accuracy of the model [8]. These works inspired our approach as our dataset contained already extracted features from satellite imagery, and our problem falls under the domain of multi-target regression.

Another related work by Li et al. proposed a hierarchical framework based on chain regression models for affective recognition from vocal bursts [7]. Although this work focuses on a different application, it shares similarities with our approach in the use of chain regression models. However, our approach differs by using a greedy approach to optimize the order of target variables, while Li et al. used a hierarchical structure to consider multiple relationships between emotional states, cultures, and emotion spaces.

In summary, our work leverages the strengths of existing approaches to feature extraction and multi-target regression while introducing a novel method for optimizing the order of target variables in the regression chain. This approach addresses the unique challenges of our task and contributes to the broader body of research on multi-target regression and satellite imagery data analysis.

3 Data

In this subsection, we will explore the various datasets that form the backbone of our project. These datasets, each with its unique characteristics and information, have been instrumental in shaping our approach and methodology.

We will delve into the details of each dataset, discussing their origins, the type of data they contain, and how they have been utilized in our project

3.1 Google Earth Engine Dataset

The dataset used in this project is a comprehensive compilation of data from Demographic and Health Surveys (DHS) [1] conducted in 58 countries, combined with various satellite imagery sources. The dataset, both training and test sets, was meticulously curated by the Geldsetzer lab at Stanford Medicine.

The primary component of the dataset is the `gee_features.csv` file. This file contains extracted features from Google Earth Engine (GEE) [13] and keys to match it with other types of data, such as country names (DHSCC), cluster numbers (DHSCLUST), and year of survey (DHSYEAR). It's important to note that certain column names in this file are not predictive features but rather identifiers and metadata. The predictive features are extracted from different sources of satellite images and other public data.

The dataset encompasses 11,959 numerical features for 120,984 DHS community reports, representing a total of 58 countries. These features include numerical meteorological and geographical data for each DHS community from Google Earth.

The `training_label.csv` file serves as the label dataset for the training set. It can be linked to the features in `gee_features.csv` using the DHSID. The project aims to predict six key health indicators: Mean BMI, Median BMI, Unmet Need Rate, Under5 Mortality Rate, Skilled Birth Attendant Rate, and Stunted Rate.

The distribution of the DHS survey points across different countries is visualized in the heatmap shown in Figure 1. This heatmap provides a global view of the number of data points available per country. The color intensity in each country corresponds to the number of data points available, with darker colors indicating a higher number of data points. This visualization helps to highlight the geographical distribution of the data and reveals the countries with the most comprehensive data coverage. It's important to note that the distribution of data points is not uniform across all countries, which reflects the varying availability and accessibility of health data in different regions. This uneven distribution poses a challenge for the prediction models, as countries with fewer data points may not be as accurately represented in the model's predictions.

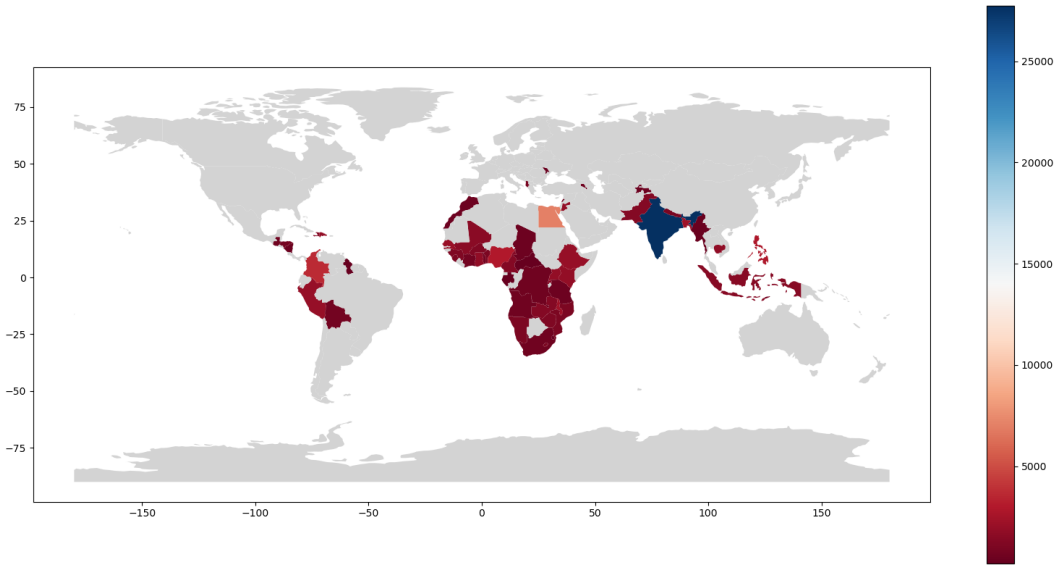


Figure 1: DHS Survey Points per Country

3.2 Training labels

The `training_label.csv` file serves as the label dataset for the training set. It can be linked to the features in `gee_features.csv` using the column `DHSID`. The project aims to predict six key health indicators: Mean BMI, Median BMI, Unmet Need Rate, Under5 Mortality Rate, Skilled Birth Attendant Rate, and Stunted Rate. Figure 1 shows a glimpse of how data looks in `training_labels.csv`.

The labels in our dataset represent key health indicators derived from the Demographic and Health Surveys (DHS). These indicators provide valuable insights into the health status of populations in low- and middle-income countries. The labels are as follows:

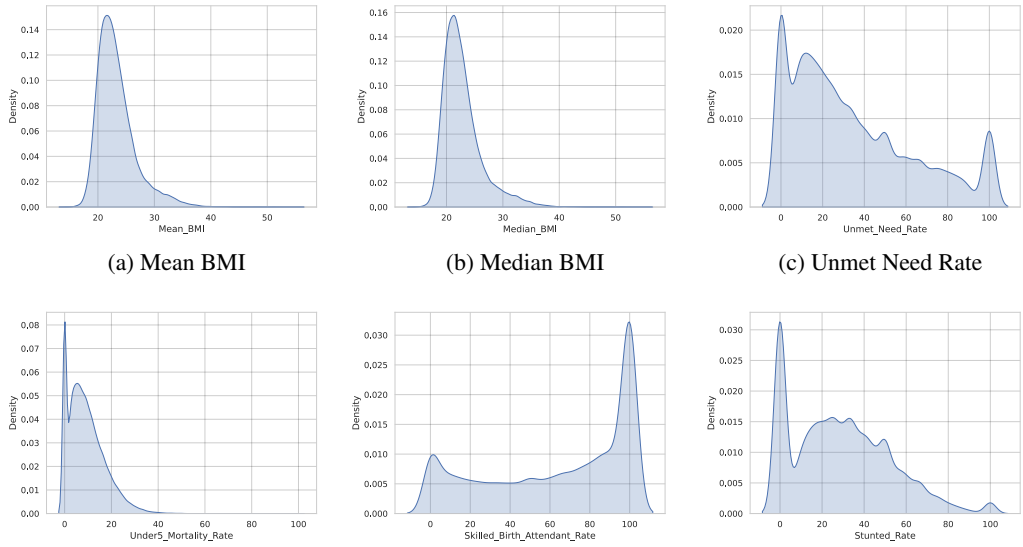
- **Mean_BMI:** This represents the average Body Mass Index in a given community. It is a measure of body fat based on height and weight that applies to adult men and women.

Table 1: Training Labels

Mean BMI	Median BMI	Unmet Rate	Under5 Mortality	Skilled Birth Attendant	Stunted Rate
24.12	25.28	50.0	9.68	100.0	20.0
23.04	21.98	7.69	8.33	66.67	0.0
26.74	26.57	7.69	2.86	100.0	0.0
27.58	28.08	0.0	9.52	NaN	0.0
24.23	23.77	20.0	23.81	50.0	0.0

- **Median_BMI:** This is the median Body Mass Index in a given community, providing a measure that is not skewed by outliers.
- **Unmet_Need_Rate:** This represents the percentage of women who want to stop or delay childbearing but are not using any method of contraception.
- **Under5_Mortality_Rate:** This is the probability per 1,000 that a newborn baby will die before reaching age five, if subject to current age-specific mortality rates.
- **Skilled_Birth_Attendant_Rate:** This is the percentage of births attended by skilled health personnel.
- **Stunted_Rate:** This is the percentage of children under 5 years of age who suffer from stunting (low height-for-age).

We have generated plots for each label, showing the label value versus the number of instances. These plots provide a visual representation of the distribution of each health indicator in our dataset. The distribution of these health indicators is shown in Figure 2.



(a) Mean BMI (b) Median BMI (c) Unmet Need Rate

(d) Under5 Mortality Rate (e) Skilled Birth Attendant Rate (f) Stunted Rate

Figure 2: Label value versus number of instances for each health indicator

As we see from the plots, the plots of Mean_BMI and Median_BMI follow a nearly Gaussian distribution, but other plots have uneven distribution with spikes at certain values. This calls for further investigation of the training labels and the application of methods such as log transformation to make the labels more normal in nature. Having a normal distribution generally helps improve the results in regression algorithms.

Also, on further analysis, we discovered that the labels had missing values at random. Figure 3 depicts the amount of missing data in each column.

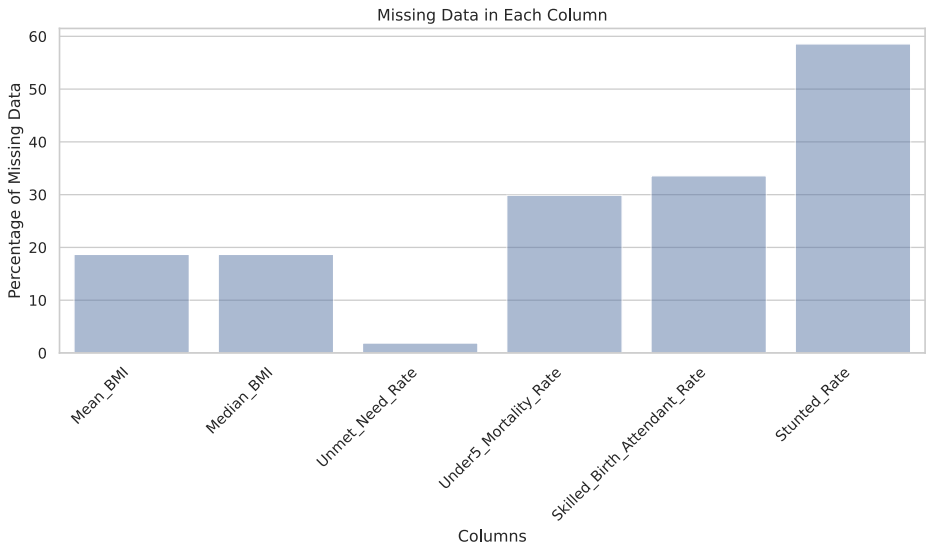


Figure 3: Amount of missing data in each column.

The column Stunted_Rate has around 57 percent values missing. Thus, we used various techniques to impute the data. The straightforward way is to delete all rows which have null values (which does give the best result for our model), but we did implement imputation techniques like mean, median imputation, KNN (K-nearest neighbours) Imputation, MICE (Multiple Imputation by Chained Equations) to check the corresponding errors in each of the imputation methods. In our final model, we have dropped the rows which have the target columns as missing and KNN imputed the rows where the covariates were missing.

3.3 World Bank Data

The World Bank website hosts a comprehensive repository of economic, financial, and social indicators collected and maintained by the World Bank, a global financial institution. This data is crucial for researchers, policy makers, and governments, as it provides invaluable insights into global development trends and economic indicators of countries.

The World Bank provides an API that allows developers to access and retrieve data from its extensive database of economic, financial, and social indicators. We utilized this API to access the indicators' data based on the countries and the year. We created an automated data retrieval script that interfaces with the World Bank API, allowing us to seamlessly obtain the relevant data for each DHSID in our training data.

The data was queried using the `wbdata` API (Python module) by feeding in the country and the year. The retrieved data was then transformed and stored in a dataframe, with DHSID used as the key to merge with the initial data. The data was retrieved only for unique Country_Code - DHSYEAR pairs, and then appended to the dataframe for all the countries.

The workflow for retrieving this data was designed to save progress periodically (written to a pickle file), allowing the process to be restarted from the last saved checkpoint if data retrieval is stopped or fails.

The data from `gee_features`, and `wbdata` was merged based on the unique columns named - `DHSID`, creating a new inclusive dataset. Our further analysis is based on this new merged dataset.

3.4 MOSAIKS Satellite Imagery Data

In an effort to further enhance our dataset and potentially improve the predictive power of our models, we tried exploring the use of MOSAIKS satellite imagery dataset. This dataset provides 4000

precomputed features about geospatial information of global land areas, which could be instrumental in understanding and predicting health indicators.

The MOSAIKS data is structured as a global grid with a resolution of 0.01 x 0.01 degrees, derived from Planet imagery. Due to the vast size of the complete dataset, which spans multiple terabytes, we strategically requested custom subsamples of the imagery that align with our project's needs. We supply a list of locations of interest via the "File Query" tool, and the MOSAIKS API allocates each input latitude and longitude coordinate to the nearest point on the global grid.

The dataset was spatially merged, using geopandas module, it was done by creating a grid of 0.005 degrees all around the grid, and then merged on the basis of intersection. All the data points couldn't be very accurately mapped to each corresponding one in the gee features dataset as the granularity of MOSAIKS data was 0.01 x 0.01 degrees, and when the latitude longitude doesn't exactly match, MOSAIKS returns the value for the nearest grid point.

After spatial merging, we moved on to use this data for training our model (we used our baseline XGBoost regressor using the multi output regression wrapper) to assess the usefulness of this data. The final scores using the MOSAIKS data can be found in Table 2.

Metric	Value
Mean_BMI	2.1570699
Median_BMI	2.2421753
Unmet_Need_Rate	19.1281003
Under5_Mortality_Rate	4.6978377
Skilled_Birth_Attendant_Rate	21.9707315
Stunted_Rate	17.6362639
MCRMSE	11.1721123

Table 2: MOSAIKS Data Results

However, we were not able to accurately map all the data, resulting in a lot of missing data for features as well as the sample prediction. With over 1,000 data points missing in sample predictions after spatial merging, and over 6,000 data points missing in the training data, there was a significant amount of entirely missing data. Even though the standalone MOSAIKS results look promising, we need to come up with more refined and better-tuned ways to spatially merge the data. Then it can be integrated with GEE features data for even better predictions. This challenge represents a potential future avenue for improving the model.

3.5 Handling Large Datasets

The initial challenge in our project was managing multiple large datasets, which exceeded memory limits. To address this, we implemented two strategies. First, we converted the dataset from CSV to Parquet format. This conversion significantly improved memory efficiency and enabled faster data access and processing. Second, we downcasted the variables to the lowest possible datatype without losing precision. This further reduced the memory footprint of the dataset and facilitated efficient data handling.

4 Data Preprocessing Pipeline

Data preprocessing is a crucial step in any machine learning project. It involves cleaning and transforming raw data into a format that can be easily understood and utilized by machine learning algorithms. In this section, we will discuss the various preprocessing techniques we have employed to prepare our dataset for model training.

4.1 Feature Selection

Selecting the optimum number of features, as well as the most relevant features, is crucial for building an efficient model. For this purpose, we trained a boosting model (XGBoost) using a multi-target regression wrapper on all the merged features. We then computed the feature importance for each feature. Evaluation was done using k-fold cross-validation.

Optimum Number of Features The optimum number of features was selected based on the graph of MCRMSE (Mean Column-wise Root Mean Square Error) score (k-fold mean) vs the number of features. Based on the graph (Figure 4), we observed a slight elbow at the number of features = 350, so we chose it as the optimum number of features.

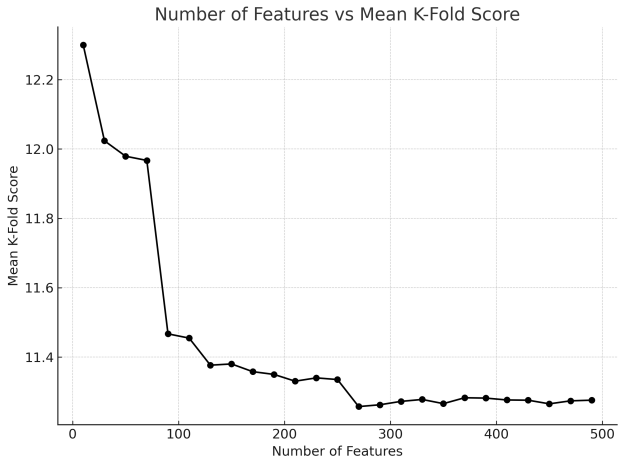


Figure 4: MCRMSE vs Number of Features

4.2 Handling Missing Values

A significant challenge in our dataset was the presence of missing feature and label data. To address this, we first dropped the data which had missing label data, as these instances lacked the necessary ground truth for training and evaluation.

Next, we addressed the remaining missing feature values through the application of the K-nearest neighbors (KNN) imputation technique. KNN imputation estimates missing values based on the values of neighboring data points. We set the number of neighbors parameter to **5**, a value determined based on the dataset's characteristics. We had also tried hierarchical imputation where the imputation was performed country-wise, ensuring that the imputed values were contextually relevant and accurate. However, KNN imputation of the entire data with k neighbors = 5 performed well, so we continued with it. The $k = 5$ was decided after comprehensively trying out various k neighbors values. Figure 5 shows the k neighbors value (x-axis) vs the MCRMSE score (on the y-axis).

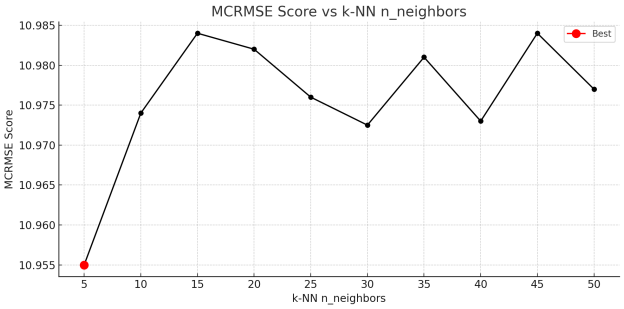


Figure 5: KNN neighbors vs MCRMSE

Finally there were still some NaN values, which were imputed using the median of the data.

4.3 Country-wise Segregation

To facilitate regional analysis, we assigned each country in the dataset to a specific region based on predefined country sets representing distinct geographical areas. These regions include East Asia

& Pacific, South Asia, Central Asia, North Africa & Middle East, Sub-Saharan Africa, Europe & Central Asia, and Latin America & Caribbean.

We implemented this by creating mappings between country codes and unique numerical identifiers, as well as between regions and their corresponding numerical identifiers. This allowed for efficient representation of countries and regions using numeric values. Additionally, we performed necessary data adjustments, such as replacing certain values in the dataset.

By applying this process to each row in the dataset, we assigned the appropriate target region to each country. The resulting region information was stored in a dedicated column, facilitating subsequent regional analyses.

This categorization enabled us to analyze countries based on their respective regions, providing valuable insights into regional patterns and characteristics. It enhanced our understanding of the data and enabled us to derive meaningful conclusions about the health indicators and associated factors within different regions.

4.4 Train-Test Split

In order to develop a model that generalizes well across different countries, it is crucial to have a representative sample from each country in the training, development (dev), and test sets. To achieve this, we employed an approach for creating the train/dev/test split at the country level.

The `split_data_country_wise(df)` function is used to perform the split. This function operates as follows:

1. **Split Data for Each Country**: For each unique country in the dataset, the data is split into three subsets:
    - *Training Set*: 80% of the country-wise data
    - *Development Set*: 10% of the country-wise data
    - *Test Set*: 10% of the country-wise data
2. **Concatenate All Sets**: After splitting the data for each country, all the training sets from each country are concatenated to form the final `X_train` set. Similarly, all the development sets are concatenated to form the final `X_dev` set, and all the test sets are concatenated to form the final `X_test` set. The corresponding labels are concatenated to form `y_train`, `y_dev`, and `y_test` sets.

This approach ensures that each set (train, dev, test) has representation from all the countries in the dataset. This is important because it helps to:

- **Mitigate Bias**: Ensuring that the model is exposed to data from all countries during training, which helps in mitigating biases towards any specific country.
- **Improve Generalization**: Providing a more comprehensive and representative sample for our machine learning models, which helps in improving the model's ability to generalize across different countries.

Ultimately, this method of splitting the data helps in developing a model that is robust and performs well on data from all the countries included in the study.

5 Experiments

5.1 Evaluation method

The primary evaluation metric used in this project is the Mean Column-wise Root Mean Squared Error (MCRMSE). This metric provides a measure of the average error of our model's predictions across all target variables. The Root Mean Squared Error (RMSE) for each target variable is calculated as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{1}$$

where $\hat{y}_i$ is the predicted value and y_i is the original value for each instance i. The MCRMSE is then the average of the RMSEs for each predicted column.

In addition to MCRMSE, we also utilize other evaluation metrics to assess the performance of our models. These include:

1. **Coefficient of Determination (R-Squared)**: This metric provides a measure of how well the variations in the predicted values can be explained by the model. It is calculated as:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2} \tag{2}$$

   where $\bar{y}$ is the mean of the original values.

2. **Mean Absolute Error (MAE)**: This metric calculates the average of the absolute differences between the predicted and actual values. It is less sensitive to outliers compared to RMSE and is defined as:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{3}$$

These additional metrics provide a more comprehensive understanding of the model's performance, taking into account different aspects of the prediction errors.

5.2 Experimental details

We used Optuna [2] for hyperparameter tuning. Optuna is an open-source hyperparameter optimization framework in Python. It is designed to optimize the hyperparameters of machine learning models, which is key to the efficient development of machine learning models.

The individual tuning was done for each model; XGBoost, LightGBM, and CatBoost. Optuna suggests the corresponding hyperparameters for the model. It fits a `RegressorChain` with the suggested model and hyperparameters and the best order of the labels found in the previous section on the training data. It then predicts the development data and computes the MCRMSE of the predictions. Optuna will then try to find the hyperparameters that minimize the MCRMSE.

We used a `MedianPruner` to stop the optimization of trials that do not appear promising. In the actual run, we conducted the optimization with 200 trials.

Finally, the best hyperparameters that we obtained and used are shown in Table 3.

Table 3: `Model HyperParameters`

Hyperparameter	XGBoost	LightGBM	CatBoost
Learning Rate	0.012576	0.012930	0.064524
Max Depth	9	9	9
Min Data in Leaf	8	-	8
Num Leaves	-	232	-
Feature Fraction	0.406291	0.385093	-
Bagging Fraction	0.908775	0.938216	-
Random State	1	1	1
n estimators	963	978	-
min child weight	8	-	-
colsample bytree	0.406291	-	-
subsample	0.908775	-	-
L2 leaf reg	-	-	8.376119
iterations	-	-	996
bagging freq	-	1	-

The greedy algorithm was used to determine the optimum order. The time taken for the greedy algorithm to come up with the optimum order is shown in the following table4:

Table 4: Greedy Algorithm Time

Model	Time (sec)
XGBoost	2500
LightGBM	12000
CatBoost	9000

The k-nearest neighbors (k-NN) algorithm was used for imputation. The imputation time taken is `2 hours, 6 minutes, and 2 seconds`.

The ensemble regression chain was used for model training. The training time taken is `1 hour, 31 minutes, and 11 seconds`. All the codes were run on a Kaggle notebook which has a configuration of 32GB of RAM and four cores of CPU. It also has access to a GPU (GPU T4*2 and GPU P100) which was used at times for tuning and training purposes.

6 Approach

We explored a variety of approaches, focusing here on the ones that yielded the best results among our submissions. Initially, our 'baseline' model was an XGBoost algorithm [5] trained with default parameters for multi-target regression on the dataset. We chose XGBoost as it excels in handling tabular data and often outperforms deep learning techniques[12], such as neural networks, achieving a score of 12.27. Subsequently, we conducted further experiments, incorporating algorithms like the TabNetRegressor[4], a deep learning-based regression algorithm implemented using the PyTorch library. Additionally, we employed other multitarget regression algorithms, including CatBoost [11] and LightGBM regressors [6]. In pursuit of enhanced performance, we also explored ensemble modeling. Specifically, we assembled an ensemble of three machine learning models: XGBoost, LightGBM, and CatBoost. These models were chosen for their strong track record in handling structured data and their efficiency in managing high-dimensional datasets. According to [9], ensemble models offer the advantage of higher accuracy and reduced overfitting.

Our current best approach (part of which is inspired from the paper [3]) employs an ensemble of three models: XGBoost Regressor, LightGBM Regressor, and CatBoost Regressor. The ensemble model combines the predictions of these three models using weights determined through Bayesian Model Averaging (BMA). BMA is a technique that calculates the weights of each model in the ensemble based on their performance. The better a model performs, the higher weight it is assigned.

In our approach, we calculate the BMA weights based on the inverse of the validation scores, meaning models with lower validation scores (indicating better performance) are assigned higher weights. The weights are then normalized so that they sum to 1.

$$w_i = \frac{\frac{1}{\text{Validation Score of Model } i}}{\sum_{k=1}^{N} \frac{1}{\text{Validation Score of Model } k}}$$

Where N is the total number of models in the ensemble.

We use the top 350 features for our models. The order of training is decided so that the later trained models benefit from the earlier predictions that have been used as features. This is achieved through a technique known as regression chaining, where the output of one model is used as an input feature for the next model in the chain. This approach allows us to capture complex dependencies between the target variables and improve the overall predictive power of our models, as most of the target variables are very closely related and belong to the same domain of Maternal and Child Health.

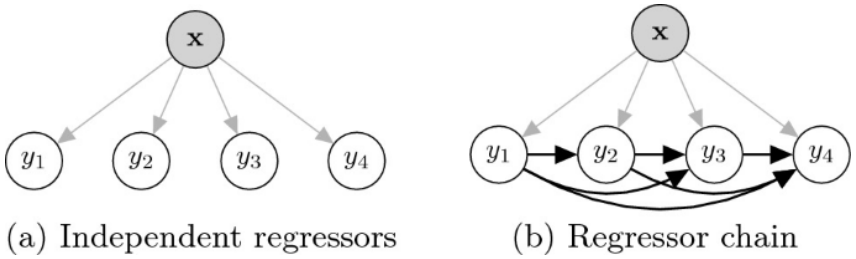


Figure 6: Independent Regression Models vs Regression chained Models (Figure Courtesy : [3])

6.1 Chaining Order

The most crucial aspect of our approach is the chaining order, i.e., the order in which the models are trained. Initially, the order was chosen based on the understanding of the relationships between the health indicators. However, it is computationally expensive to evaluate all possible 6! or 720 orders. Hence, a **novel method** was devised by our team member Vedant Zope and implemented by him, which employs a greedy approach to find the optimal order for our model. This method significantly reduces the computational burden by requiring the evaluation of only 21 orders (6+5+4+3+2+1) to find an optimal order.

The greedy approach works as follows:

- For each label, calculate the MCRMSE by fixing the already selected labels and adding the current label to the order.
- Select the label with the minimum MCRMSE and add it to the current order.
- Remove the selected label from the list of all labels.
- Repeat the above steps until all labels are added to the order.

Pseudo code for the approach is given below:

```
Procedure FullGreedySearch:
  Inputs: model, X_train, y_train, X_dev, y_dev
  all_targets <- list of all target variables
  current_order <- empty list
  history <- empty list
  while all_targets is not empty do:
    best_score <- infinity
    best_target <- null
    for each target t in all_targets do:
      temp_order <- current_order + t
      score <- EvaluateOrder(model, temp_order, X_train, y_train, X_dev, y_dev)
      add (temp_order, score) to history
      if score < best_score then:
        best_score <- score
        best_target <- t
      end if
    end for
    print "Added target", best_target, "to the order. Current best score:", best_score
    add best_target to current_order
    remove best_target from all_targets
  end while
  return current_order, history

Procedure EvaluateOrder:
  Inputs: model, order, X_train, y_train, X_dev, y_dev
  full_order <- order + all remaining targets not in order
  chain <- create a RegressorChain with model and full_order
  fit chain on X_train, y_train
```

```
y_pred <- predict X_dev using chain
score <- calculate MCRMSE between y_dev and y_pred
return score
```

The actual implementation of the algorithm can be found in our GitHub repository, the link to which is provided at the end of this report.

In summary, the greedy approach efficiently finds a good chaining order for our models despite the computational limitations, significantly impacting the model's final performance.

7 Results

The results of our models are presented in the table below 5. The table shows the Mean Column-wise Root Mean Squared Error (MCRMSE) for each model, which is the metric we used to evaluate our models' performance. Lower MCRMSE values indicate better performance.

Index	Model/Approach	MCRMSE
1	Tabnet Regressor	12.60
2	Country Wise XGBoost models	12.27
3	Fully Connected Neural Network Regression	12.249
4	CatBoost top 120 features	12.263
5	Ensemble Regression Chain(+MICE Imputation)	12.18
6	Ensemble (+PCA)	12.12
7	XgBoost top 120 features	12.004
8	Tensorflow Decision Forests	11.38
9	Ensemble Regression Chain(+KNN Imputation)	11.36
10	Multioutput Stacking Regressor	11.201
11	Ensemble Regression Chain	11.2
12	hyperparam Tuned Ensemble Regression Chain	11.02
13	order Tuned Ensemble Regression Chain	10.99
14	individual ensemble order tuned Ensemble Regression Chain	10.95
15	k fold tuned (order+hyperparam) Ensemble Regression Chain	10.9173

Table 5: Model Performance

Our fine-tuned Ensemble Regression Chain model performed the best, with an MCRMSE of 10.9173(2nd position on leaderboard). This model leverages the power of ensemble learning and regression chaining, which allows us to capture complex dependencies between the target variables and improve the overall predictive power of our models.

We have employed individual tuning for each of the ensemble models. It calculates the optimum order (using a greedy algorithm) for each of the models and also calculates the optimum hyperparameters (using Optuna). Finally, we use Bayesian Model Averaging (BMA) to calculate the final prediction. This results in a very robust model, as the predictions are averaged over various chaining orders and different models.

Overall, our results suggest that our approach of using ensemble learning and regression chaining is effective for predicting health indicators from satellite imagery and other geotagged data sources. However, there is still room for improvement, particularly in the areas of additional data and feature engineering. Future work could explore more data, feature selection methods, and model architectures to further improve performance.

Additionally, as can be seen from the plot 7, our approaches are getting better over time. This indicates that the strategies and techniques we applied in the course of the project helped us improve our models and their predictions. This is a positive indication that our strategies were effective and will serve as a solid foundation for further improvements and developments.

8 Conclusion

In this project, we have explored the potential of machine learning and satellite imagery to predict key health indicators in low- and middle-income countries. Our approach has demonstrated promising

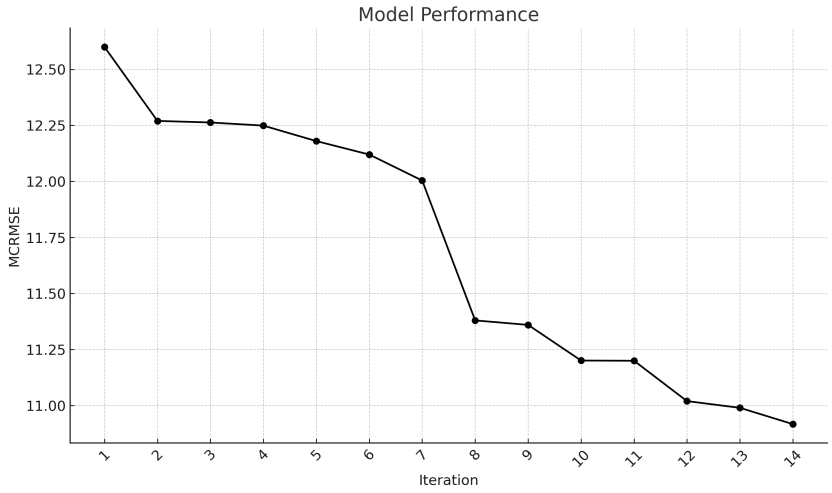


Figure 7: Model Performance vs Iterations

results, with an ensemble of three powerful boosting algorithms: XGBoost, LGBMRegressor, and CatBoostRegressor, achieving an impressive score of 10.9173.

Key highlights of our work include:

- **Regression Chaining:** One of the strengths of our approach is the use of regression chaining, which allows us to take into account the interactions among the target variables. This is particularly important as the health indicators we are predicting are not independent of each other. By incorporating the predictions of earlier models as features in subsequent models, we are able to capture these complex dependencies and improve the overall predictive power of our models.
- **Greedy Approach for Chaining Order:** We employed a greedy algorithm to determine the optimal order for regression chaining. This involved calculating the error for each possible order and selecting the one with the lowest error. By doing so, we were able to find the order of target variables that resulted in the best performance for our model, further enhancing its predictive power.
- **Integration of Additional Data:** We have already explored the MOSAIKS satellite image data, but a more efficient plan is needed for spatial merging due to the resolution of the data. We have also used World Bank data as additional data, and we have seen that it has increased the accuracy of the model.

In conclusion, our project represents a significant step forward in the application of machine learning and satellite imagery for health monitoring in low- and middle-income countries. We believe that there is a lot of scope for further refinement and exploration of advanced techniques, which can unlock even greater improvements in prediction accuracy and contribute to the global health monitoring efforts.

9 Contributions

Vedant Zope: Worked on additional data retrieval workflow, developing the ensemble regression chain model, and came up with the greedy approach for optimal order determination. Also worked on developing the report in LaTeX.

Soham Tripathy: Worked on preprocessing, trying and tuning various models, and feature selection methods. Also contributed to the report content and presentation preparation.

Kushagra Parmeshwar: Worked on conducting literature review, handling null values, preprocessing, tuning the workflow, and report development in LaTeX.

References

[1] The dhs program - quality information to plan, monitor and improve population, health, and nutrition programs.

[2] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631, 2019.

[3] Ekaterina Antonenko and Jesse Read. Multi-modal ensembles of regressor chains for multi-output prediction. In Tassadit Bouadi, Elisa Fromont, and Eyke Hüllermeier, editors, *Advances in Intelligent Data Analysis XX*, pages 1–13, Cham, 2022. Springer International Publishing.

[4] Sercan Ö Arik and Tomas Pfister. Tabnet: Attentive interpretable tabular learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 6679–6687, 2021.

[5] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2016.

[6] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qi Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30*, pages 3146–3154. Curran Associates, Inc., 2017.

[7] Jinchao Li, Xixin Wu, Kaitao Song, Dongsheng Li, Xunying Liu, and Helen Meng. A hierarchical regression chain framework for affective vocal burst recognition. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023.

[8] Saulo Martiello Mastelini, Victor Guilherme Turrisi da Costa, Everton Jose Santana, Felipe Kenji Nakano, Rodrigo Capobianco Guido, Ricardo Cerri, and Sylvio Barbon. Multi-output tree chaining: An interpretative modelling and lightweight multi-target approach. *Journal of Signal Processing Systems*, 91:191–215, 2019.

[9] David Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *Journal of artificial intelligence research*, 11:169–198, 1999.

[10] Simone Piaggesi, Laetitia Gauvin, Michele Tizzoni, Ciro Cattuto, Natalia Adler, Stefaan Verhulst, Andrew Young, Rhiannan Price, Leo Ferres, and André Panisson. Predicting city poverty using satellite imagery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 90–96, 2019.

[11] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems 31*, pages 6638–6648. Curran Associates, Inc., 2018.

[12] Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90, 2022.

[13] Google Earth Engine Team. Google earth engine: A planetary-scale geo-spatial analysis platform, 2015.