Monitoring Maternal & Child Health in Low- & Middle- Income Countries using Geo-tagged Data

Stanford - IIT Kharagpur Project

Abhishek Singh
Department of HSS
IIT Kharagpur
abhishekks.iitkgp@gmail.com

Supreet Sahu
Department of ECE
IIT Kharagpur
supreetsahu02@kgpian.iitkgp.ac.in

Amruit Sahoo
Department of AI/ML
IIT Kharagpur
amruit2k@gmail.com

Abstract

Accurately predicting health indicators is a task of utmost importance, when it comes to understanding the well-being of a society. Especially in low- & middle-income countries (LMICs), where monitoring maternal & child health (MCH) can play a crucial role in the development and prosperity of people. In such countries, physical surveys in the past, have been able to sample only a tiny fraction of the entire nation. Our aim is to demonstrate that machine learning and deep learning when applied to geo-tagged datasets can accurately estimate key indicators to monitor maternal & child health efficiently. Our main contribution is utilizing geotagged datasets to accurately predict MCH indicators. We obtain our best results using SHAP library for feature selection and LightGBM as the main model.

1 Introduction

Death registers in most low- and middle-income countries (LMICs) are not reliable enough. As a result, maternal and child deaths in LMICs are often not accurately documented in civil registration and vital statistics. Instead, information on key indicators of maternal and child health (MCH) and the extent of coverage of essential services, such as childhood vaccinations, is primarily obtained through nationwide household surveys, for example, Demographic and Health Surveys (DHS), which are expensive & time-consuming. Further, these surveys have limitations in providing detailed insights about the variations of MCH indicators and service coverage across different communities. This is due to the fact that they typically sample only a small fraction of villages and neighborhoods in a country (usually less than 2%) and the data collected may not reflect the current situation.

With the advent of deep learning and advances in satellite data, we can accurately predict MCH indicators in real time. Correctly predicting these indicators in time will inform the nations to reach out to communities that are in need of help. The accessibility and coverage of satellite data can help us achieve our goal. Doing so with a limited amount of data is a challenging as well as a motivating task, as areas having the least health data are the ones most likely to have inadequate health infrastructure & facilities.

Our contribution is the utilization of satellite-derived geotagged datasets to accurately predict MCH indicators. For this we use the MOSAIKS dataset as a baseline & Google Earth Engine (GEE) Features as our primary dataset. To achieve better performance with limited data we engineered or selected features using Principal Component Analysis (PCA), Spearman Co-relation, Tree-based models & SHAP library. Finally, we experiment with these features on an extensive set of machine learning & deep learning architectures. We found that label-wise feature selection using SHAP library with LightGBM model performs the best with a score of **10.95156**.

2 Related Work

Rolf et al [1]. obtain a task-agnostic high-dimensional feature representation of satellite images with their **Multi-task Observation using Satellite Imagery and Kitchen Sinks (MOSAIKS) model**. The authors demonstrate that these features generalize across 9 diverse prediction tasks such as forest cover, house price, and road length etc. They further show that these high-dimensional features can be used with simple Linear Regression models to achieve comparable results with deep learning models like CNN & ResNet-18, at a fraction of computational cost.

Global health monitoring from satellite data through multi-task learning by Nangi et al [2] addresses the effectiveness of Multi-task deep learning on satellite images and features to predict health factors. They designed regression models based on various datasets like SustainBench and resources like NASA Landsat and Google Earth images. The paper mentions that different health parameters require different model architectures. They also found that proper feature reduction and selection further helped the model to accurately predict.

Grinsztajn et al [3] investigate the reasons behind the superiority of tree-based models over deep learning techniques in **Why do Tree based models still outperform deep learning on typical tabular data?**. They also showcase the importance of feature selection for the performance of tree-based models and the robustness of such models to redundant features.

While there has been progress in the use of machine learning & deep learning models for satellite imagery, there is not much work that explores its effectiveness in monitoring health on a global level, which is our main objective.

3 Approach

3.1 Data Preprocessing

Data preprocessing is a critical stage in the data science pipeline, serving as the bridge between raw data and machine learning models.The quality and structure of the data directly influence the performance and interpretability of the predictive models. Preprocessing steps help in transforming the raw data into a format that can be readily consumed by machine learning algorithms. Given the experimental nature of our project, we employed multiple machine learning models to identify the best-performing one.

Each of these models comes with its own set of assumptions and requirements for the data, necessitating unique preprocessing steps. This section aims to provide a comprehensive overview of the general and model-specific preprocessing tasks carried out. By detailing these steps, we ensure the replicability of our results and offer insights into the intricacies involved in preparing data for diverse machine learning models.

Data cleaning is a foundational step in our data preprocessing pipeline, ensuring the integrity and quality of the dataset before it undergoes further transformations. This essential phase involves the identification and resolution of various anomalies and inconsistencies, such as

1. **Dropping Non-Predictive Features**

   We identified 21 columns that didn't contribute to any meaningful or relevant information to our target variable. These non-predictive columns, are 'new_ind', 'index', 'ADM1DHS', 'ADM1FIPS','ADM1FIPSNA', 'ADM1NAME', 'ADM1SALBCO', 'ADM1SALBNA', 'ADM1SALNA', 'ALT_GPS', 'CCFIPS', 'DATUM', 'DHSCC', 'DHSCLUST', 'DHSYEAR', 'F21', 'F22', 'F23', 'SOURCE', 'ZONECO', 'ZONENA'. We made this decision based on variance and the number of missing values in the columns. These columns have a large number of missing values or have little to no variance.

2. **Handling of Missing Values**

   We first removed columns that had more than 50 percent missing values since such a large proportion of missing data can distort the predictive model and reduce its

accuracy. Then impute the missing values with median. The median was a strategic choice because it's less affected by outliers and provides a more robust measure of central tendency.

3. **Handling Duplicates**

   Data has 4,711 duplicate DHSIDs which need to be removed before putting into the model. We have taken all rows that have the same DHSID and combined them into one row by taking the maximum value of each column using groupby('DHSID').max() function in pandas.

4. **Data Splitting**

   Then the data is split into a ratio of 90:10 into a training and validation set before doing other preprocessing steps to reduce the data leakage problem.

5. **Encoding Categorical Variables**

   Categorical variables usually need to be one-hot encoded for neural networks when their categories are independent of each other. In our data 'key1', 'URBAN_RURA', and 'DHSREGNA' are the categorical variables. One-hot encoding avoids creating an arbitrary ordinal relationship between different categories, which can happen with label encoding. It ensures that the neural network model treats each category independently.

   Whereas Tree-Based Models can handle categorical variables without one-hot encoding. Hence categorical variables such as 'key1', 'URBAN_RURA', and 'DHSREGNA' is label encoded for the tree based models.

6. **Extra Pre-processing steps for Neural Networks**
   (a) **Outliers:** We applied a technique called Winsorizing, which involves setting thresholds at the 5th and 90th percentiles due to the presence of bigger outliers. The idea is to limit the influence of these extreme values without completely discarding them. By adjusting outliers to fall within the specified percentile limits, we maintained the overall distribution of the data while reducing the impact of those extreme values.

   (b) **Feature Scaling:** The Standard Scaler is a widely-used feature scaling method that transforms each feature by removing the mean and scaling it to unit variance. This technique is particularly beneficial for algorithms that are sensitive to the magnitude of features, such as Neural Networks. .

3.2 Feature Engineering

Given the complexity and dimensionality of our dataset, it becomes crucial to identify the most informative features that contribute meaningfully to predictive accuracy. By employing various selection methods we aim to distill the most relevant features from the original dataset. This strategic reduction not only enhances model interpretability but also mitigates the risk of overfitting, thereby creating a more robust and generalizable machine learning model.

1. **PCA**

   The basic idea behind PCA is to capture the maximum amount of variance in the data using a smaller number of dimensions. The principal components are constructed in such a way that the first component accounts for the highest possible variance, followed by the second component, and so on.

By plotting the cumulative explained variance against the number of dimensions, we can observe how quickly the explained variance accumulates. We have chosen 99% as a threshold to select the number of dimensions.

By applying PCA, we reduce the dimensionality of the data, eliminate redundant or highly correlated features, and simplify the representation of the dataset. This can be particularly be useful when dealing with high-dimensional data like ours.

2. **Spearman Co-relation**

We employed the Spearman correlation, a non-parametric statistical measure, to assess the strength and direction of the monotonic and non-linear association between the individual input features and the individual target variables. Spearman correlation does not assume linearity, rendering it appropriate for analyzing non-linear associations. Spearman correlation relies on the assumption that the variables are measured on at least an ordinal scale. It is also robust to the influence of outliers.

We extracted the top 3000 features from the original gee features of DHS survey by using the F-Score of Spearman correlation as the evaluation criteria.

3. **Random Forest Feature Selection**

Conventional feature selection methods often rely on statistical tests or domain knowledge, which might overlook intricate relationships present within the data. Random Forest, an ensemble learning algorithm comprised of multiple decision trees, offers an alternative approach by utilizing its built-in feature importance metrics. The algorithm measures the reduction in impurity (e.g., Gini impurity) achieved by including a specific feature across all decision trees. This yields importance scores that highlight each feature's contribution to predictive performance.

We calculated the feature importance scores of all the predictive features and sorted them in the decreasing order of their scores. Then we experimented by taking different numbers of top features (e.g., 100,200,500,1000, etc.) and trained the model on these features.

4. **LightGBM Feature Selection**

LightGBM is a gradient-boosting framework that employs a histogram-based approach for tree construction. By harnessing its gradient-boosting capabilities and intrinsic feature importance measurement, we utilize this to uncover valuable insights into the role of features in predictive tasks. It utilizes the gain achieved by a feature when creating a split in a decision tree. This gain quantifies the contribution of the feature to improving the model's performance.

5. **SHAP Feature Selection**

SHAP (Shapley Additive exPlanations) [4], a cutting-edge technique rooted in cooperative game theory, for feature interpretation and selection. SHAP values provide insightful explanations for individual feature contributions to model predictions. It makes the whole procedure more generalized and accurate.

Even though for the above methods utilizing LightGBM or XGBoost for feature selection work well enough, but they are not specialized for the purpose of selecting features, and thus not always reliable.

SHAP is a model-agnostic technique and therefore we can use it to test with different models. SHAP values ensure that feature contributions fairly distribute the difference

between the instance's prediction and the average prediction across all possible feature subsets.

3.3 Post-Processing

1. **Dealing with 25 missing samples in test-set**

   There are 25 DHSIDs in the test set whose input features are not available in the main GEE Features Dataset. So these 25 samples cannot be predicted by the model. The only way to do this is to impute them manually. To ensure proper and accurate imputation the 25 samples are imputed based on the rest 14975 samples.

   The predictions of the 14975 samples are grouped by the parameters "DHSCC" (country) and "DHSYEAR" (year) using the Pandas GroupBy object function. Each of the 25 predictions is imputed by their respective group's (formed by Groupby) median value.

   The logic of the above process is that for a given DHSID, it's respective country and the year of the survey provide a good approximate representation of the actual health parameters of the place.

2. **Bounding range of the predicted values**

   As the model is of regression type, it can predict values above 100 (slightly above 100) and also below 0 (slightly less than 0). According to the literature, parameters like Unmet Need Rate, Stunted Rate, Under5 Mortality Rate, and Skilled Birth Attendant Rate should lie between 0 and 100. So, the values that are outside of the range are clipped to the nearest value (which can be either 0 or 100) and the final predicted values are bounded within the pre-defined range

3.4 Model architectures

We experimented with a wide variety of popular machine learning & deep learning model architectures.

1. **Linear Regression** We implement a simple linear regression model to use it as a baseline.
2. **XGBoost** We implement an extreme gradient-boosting algorithm to test the performance of tree-based models on the GEE Features Dataset.
3. **LightGBM** We implement a LightGBM model as an alternative to the XGBoost model, in pursuit of better accuracy and faster training.
4. **Neural Network** We also implement a 3-layer Neural Network with ReLU & Sigmoid activations, to test the performance of deep learning models on the GEE Features Dataset.

All the data pre-processing steps, feature engineering methods, model architectures & post-processing steps have been **coded by us** & no code is copied from any other source available on the net.

4 Experiments

4.1 Data

1. **Google Earth Engine (GEE) Features and Labels**

   The Google Earth Engine (GEE) dataset used for this machine learning project had 120,984 rows and 11,945 columns. The dataset contained diverse data types, including floats (7,227 columns), integers (4,703 columns), and objects (15 columns). Out of the features, 21 columns were identified as non-predictive and were removed during data preprocessing. Additionally, 594 columns had more than 50% missing values, which were also removed.

There are 6 Labels that are to be predicted namely *Mean_BMI, Median_BMI, Unmet_Need_Rate, Under5_Mortality_Rate, Skilled_Birth_Attendant_Rate, and Stunted_Rate*.

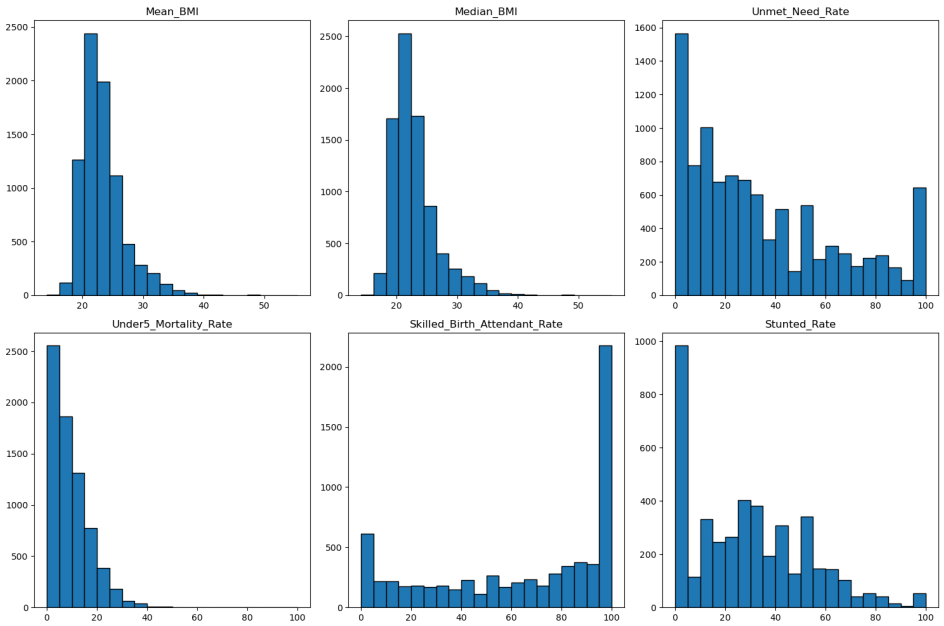


Figure 1: Histogram Distributions for each Label

	Mean_BMI	Median_BMI	Unmet_Need_Rate	Under5_Mortality_Rate	Skilled_Birth_Attendant_Rate	Stunted_Rate
% NaN	18.682904	18.682904	1.883728	29.888409	33.577165	58.567076

Figure 2: Percentage of NaN values for each Label

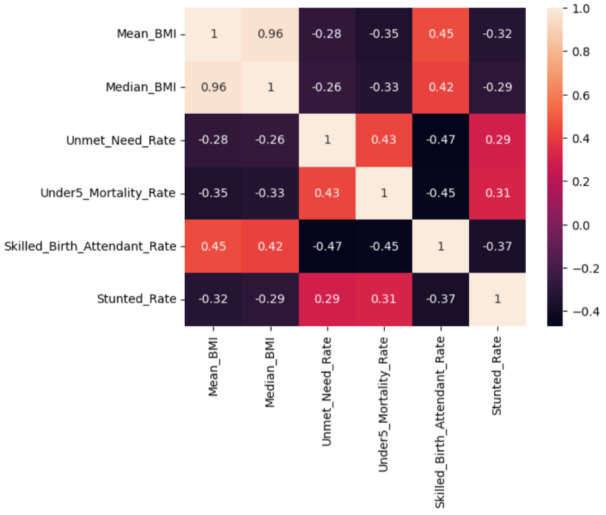


Figure 3: Heat map correlation of Labels

2. **MOSAIKS Features**

   We used the MOSAIKS API [5] to obtain 4000-dimensional feature vectors of the geotagged data points respective to their DHS IDs. MOSAIKS API has a precision of 0.005 degrees whereas the available GEE Features dataset has a higher precision of 0.000001 degrees for latitude & longitude values of a given location. So we approximate the latitude & longitude values of the location to the required precision and use these values to generate the MOSAIKS Features dataset. Approximately data for about 15% of DHS IDs was not available through the MOSAIKS API.

4.2 Evaluation method

We evaluate our findings on the basis of mean column-wise root mean squared error (MCRMSE). The final score is the average of individual RMSEs for each predicted column/label.

Root mean squared error (RMSE) is defined as:

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

4.3 Experimental details

4.3.1 Resources used

We conducted all of our experiments on the Kaggle platform. We also used Google Colab for some of the experiments. We utilized the P100 and T4X2 GPUs to speed up the training and inference of our models, and the TPU instance (300GB RAM) for steps that required high RAM, e.g. data pre-processing for GEE Features dataset, and feature selection. We were able to achieve remarkable results even with the limited resources that we had through Kaggle (30 hours of GPU per week). One can surely achieve a better score within a very short duration if the same set of experiments are performed on a powerful system. With additional resources, we could have rapidly conducted supplementary experiments and developed a more prolific model.

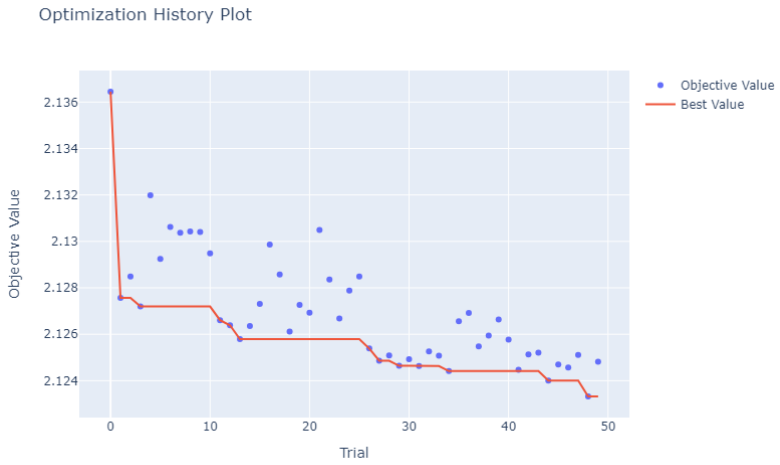
4.3.2 Model Configurations & Hyperparameters

We ensure that our model is robust & generalized to unseen data, by training with K-Fold Cross-Validation. We used early stopping criteria for 50 iterations to prevent the model from overfitting. Hyperparameter tuning for all the models is performed using the **Optuna** library with TPE Sampler as the optimization method. The following hyperparameters were tuned for their respective models

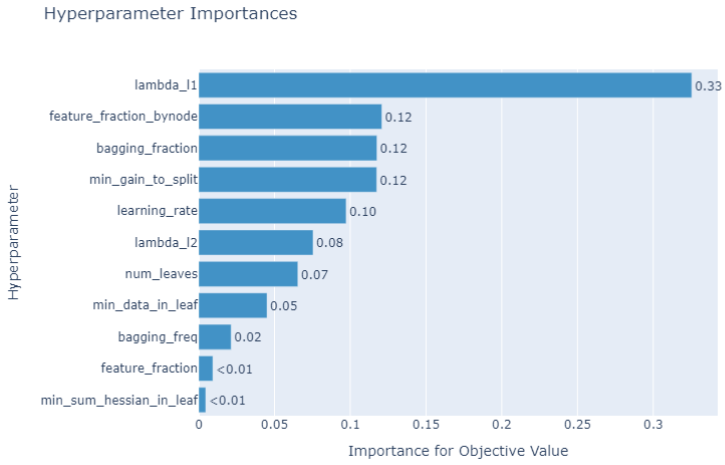
A. LightGBM Hyperparameters (as in 4): $learning\_rate$, num_leaves, $feature\_fraction$, $feature\_fraction\_bynode$, $bagging\_fraction$, $bagging\_freq$, $min\_data\_in\_leaf$, $min\_sum\_hessian\_in\_leaf$, $lambda\_l1$, $lambda_l2$, $min\_gain\_to\_split$, num_boost_rounds.

B. XGBoost Hyperparameters: *eta(learning rate)*, max_depth, $colsample\_bytree$, $colsample\_bylevel$, *subsample*, min_child_weight, *alpha, lambda, gamma and* num_boost_rounds.

C. Neural Network Hyperparameters: *number of epochs, batch size, learning rate, weight decay & dropout rate*. The Neural Network consisted of 3 layers and was trained with an RMSE loss function and Adam as optimizer.



(a) Tuning with the objective of minimizing RMSE



(b) Importance of hyperparameters

Figure 4: Hyperparameter tuning for Median_BMI label using LightGBM model

Figure 5: Scatter Plot depicting convergence on trials

4.4 Results

These are the quantitative results obtained with the models evaluated on the 20% of the 15k test dataset provided on the kaggle platform.

Datasets	Feature Selection/Engineering	Model	MCRMSE Score
MOSAIKS Features		Linear Regression (Baseline)	18.35665
		Neural Network (Baseline)	16.77518
GEE Features	PCA	Linear Regression (Baseline)	19.02568
		XGBoost	13.58995
		LightGBM	12.48749
		Neural Network	11.85509
	Spearman Co-relation	LightGBM	11.33768
	Random Forrest	LightGBM	11.04336
	LightGBM	LightGBM	11.00731
	SHAP	LightGBM	**10.95156**

Table 1: MCRMSE scores using various models with different methods for feature selection/engineering

5 Analysis

5.1 Strict training on non-NaN labels only

We train all our models only on the data points for which the training label is available. We also tried various imputation methods e.g. KNN, MICE, and basic approaches using Mean & Median to fill the missing values and create a complete dataset. However, the process of imputation negatively impacted the performance of the model as a lot of noise was introduced. Even though the training set for some of the labels becomes very small such as Stunted Rate as can be seen in 2, no noise leaks into the trained models.

5.2 Tree-based models outperform Deep Learning models

As can be seen in the results section 1, Tree-based models beat Neural Networks when tested with the same input features. This is in accordance with the findings of Grinsztajn et al [3], where the authors show the superiority of Tree-based models over Deep Learning models in the case of tabular data (like GEE Features Dataset). We also find that the Tree-based models are robust to redundant features to a certain extent.

Moreover, Tree-based models are lightweight and faster. This helped us to perform numerous experiments utilizing hyperparameter tuning and cross-validation, even with our limited resources and computing power. This would not have been possible if we had used Deep Learning models. Hence, Tree-based models can be easily hosted and used effectively in high-frequency real-time scenarios, as they consume less space & time.

5.3 Models perform badly on certain labels

We observe that our models perform relatively badly on certain specific labels: *Unmet_Need_Rate*, *Skilled_Birth_Attendant_Rate* & *Stunted_Rate*. As we can see the RMSE scores 6 for each label on the cross-validation set using LightGBM model. We can see that the model performs well on the other three labels: *Mean_BMI*, *Median_BMI* & *Under5_Mortality_Rate*.

Upon studying the distributions for each label 1, we suspect that this could be due to the fact that the bad set of labels has a skewed and imbalanced distribution whereas the good set of labels has a relatively smoother and Gaussian-like distribution. Therefore the model is able to easily learn the distributions for the good set of labels, whereas it fails in the case of the bad set.

Target variable	Cross-validation loss
Mean_BMI	1.98038
Median_BMI	2.12325
Unmet_Need_Rate	18.87339
Under5_Mortality_Rate	5.417428
Skilled_Birth_Attendant_Rate	19.33726
Stunted_Rate	18.92689

Figure 6: RMSE scores on cross-validation sets

5.4 Importance of Features Selection

Upon observing the results 1, we find there is a significant decrease in RMSE with the change in the method of feature selection. We tried a wide variety of feature selection & engineering techniques e.g. PCA, Spearman Correlation, Random Forrest, LightGBM & SHAP.

Performing numerous experiments, we found that the number of features is also an important factor for the results. We suspect that there is a certain threshold only above which the features should be taken into consideration for training the model.

5.5 Significance of Features depicting Location & Time

We find that the features depicting location & time are very important e.g. *LATNUM, LONGNUM, DHSYEAR*. We observed that these features have high importance scores for most of the labels, and therefore contributed significantly to the performance of the final models.

Our finding is also supported by this world map visualization for the labels *Stunted_Rate & Unmet_Need_Rate* in 7 & 8. We observed that locations with similar values are seen to be clustered together. A place's latitude & longitude also contain information about the weather conditions and other demographic indicators. For example, we can see that in the western-central part of Africa *Stunted_Rate* & *Unmet_Need_Rate* is higher when compared to the coastline region. We can similarly state that the feature *DHSYEAR* is equally significant, as the timeframe in a period can define the economic conditions of a nation. To look at the world map visualizations of other labels: Appendix

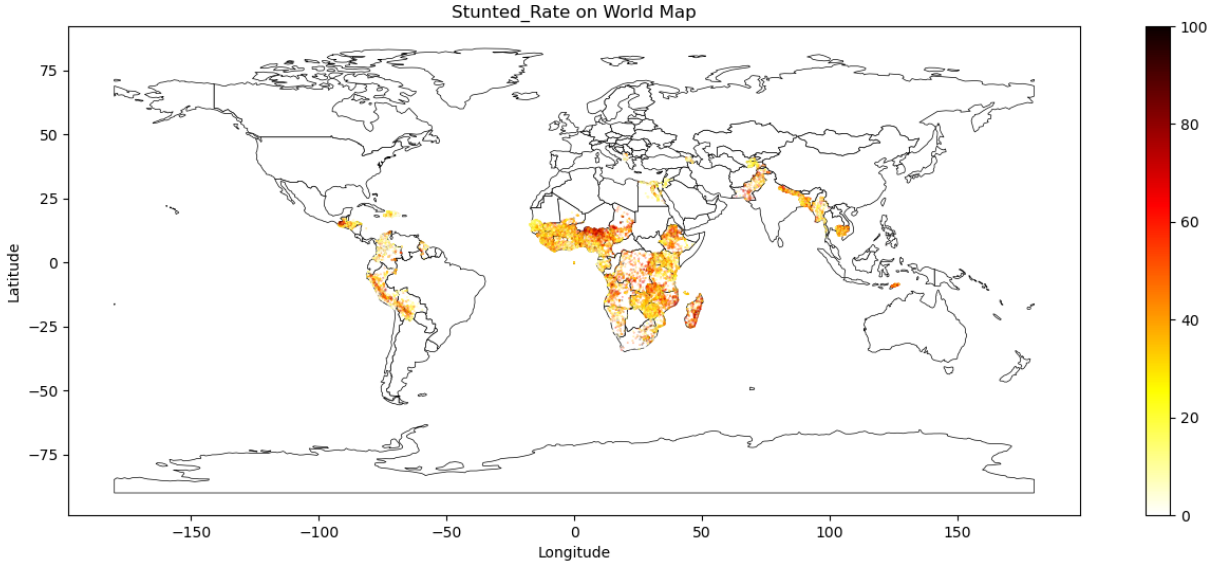


Figure 7: Stunted Rate

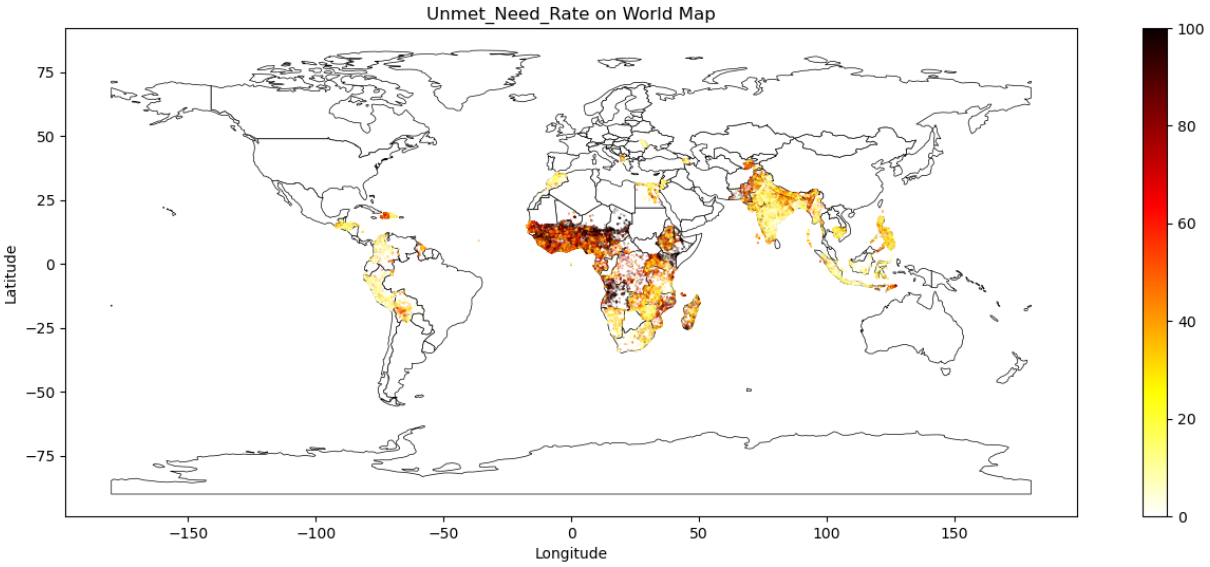


Figure 8: Unmet Need Rate

6 Conclusion

In summary, we explored various ways to accurately predict and monitor maternal & child health, by leveraging machine learning & deep learning techniques on geotagged satellite datasets. We performed various experiments to understand the effectiveness of models and the importance of feature selection. We also analyzed the results to derive intuitive insights. We conclude that the village- and neighborhood-level prevalence of maternal & child health indicators can be accurately estimated from geo-tagged data.

7 Team Contributions

- **Supreet:** My major involvement was with the Feature Selection Process and ML model selection and implementation. I prepared, implemented, and tested the XGBoost and LightGBM Models. I designed the code for these two from scratch. Further, I carried out an extensive amount of both manual and specialized hyperparameter tuning on these two models. I analyzed the input features and model hyperparameters individually on each target variable/column. This analysis significantly improved the accuracy of the models. I also assisted in the missing value analysis and imputation by providing some logical and domain-based ideas to impute missing values.

- **Abhishek:** My role revolved around researching and implementing various data preprocessing techniques. I delved into multiple methods to determine which could best enhance our dataset's quality. As a result, I proposed several data variations to serve as the training ground for our model. The most critical aspect of my contribution was the construction of a preprocessing pipeline and finding new Feature selection methods which greatly impacted our predictions. My work not only increased the efficiency of our workflow but also significantly improved the accuracy of our predictive models.

- **Amruit:** I primarily worked on tuning the LightGBM model and using SHAP to select top features. I tested the MOSAIKS API to get the baseline results. I worked on building the Neural Network architecture and performing hyperparameter tuning using Optuna to get the best results. I also did a considerable amount of research in the area of Deep Learning for Satelite Data & Images. I also worked on exploring various methods to impute the missing

values in the training labels. Like the rest of my teammates, I helped write all sections of this report, but most heavily in the introduction, related work, analysis & conclusion sections.

We would like to thank our mentors Mr. Haojie Wang and Professor Pascal Geldsetzer for this wonderful opportunity. We as a team are glad to have been able to contribute to such a novel project. We hope our work has a positive and widespread impact on communities in need.

References

[1] Esther Rolf, Jonathan Proctor, Tamma Carleton, Ian Bolliger, Vaishaal Shankar, Miyabi Ishihara, Benjamin Recht, and Solomon Hsiang. A generalizable and accessible approach to machine learning with global satellite imagery. *Nature communications*, 12(1):4392, 2021.

[2] Sharmila Nangi. Health indicators multi-task. `https://github.com/sharmilanangi/Health_Indicators_MultiTask`.

[3] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? *Advances in Neural Information Processing Systems*, 35:507–520, 2022.

[4] Fernando López. Shap: Shapley additive explanations. `https://towardsdatascience.com/shap-shapley-additive-explanations-5a2a271ed9c3`.

[5] Esther Rolf. Multi-task observation using satellite imagery kitchen sinks. `https://siml.berkeley.edu/portal/index/`.

A Appendix

A.1 World Map Visualizations

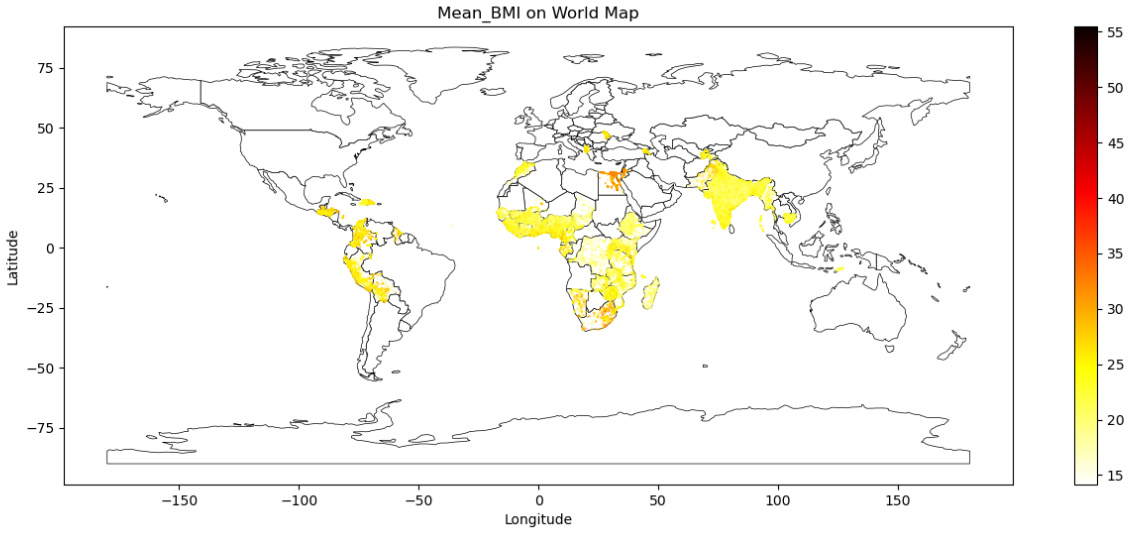


Figure 9: Mean BMI

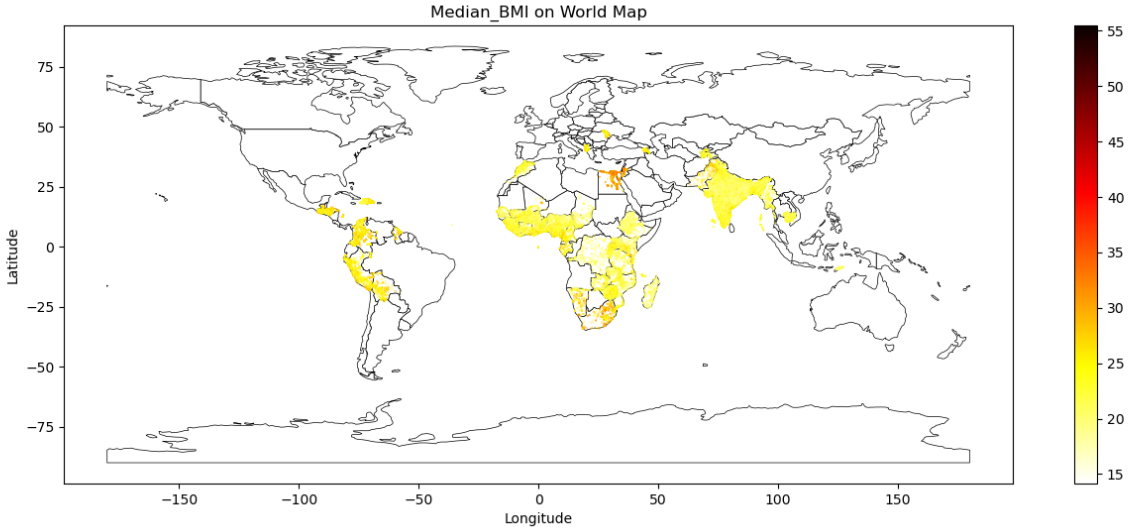


Figure 10: Median BMI

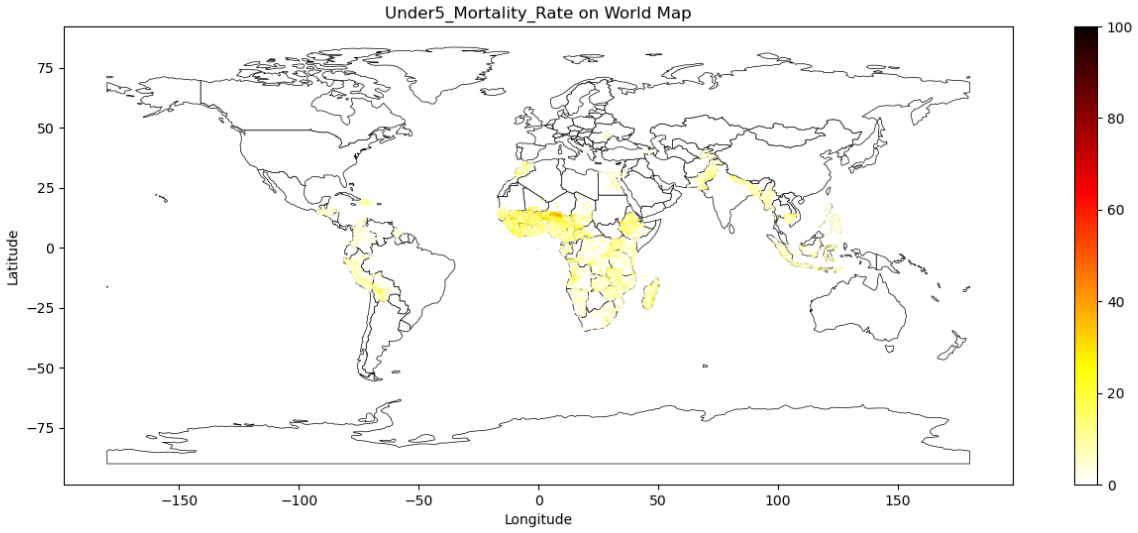


Figure 11: Under-5 Mortality Rate

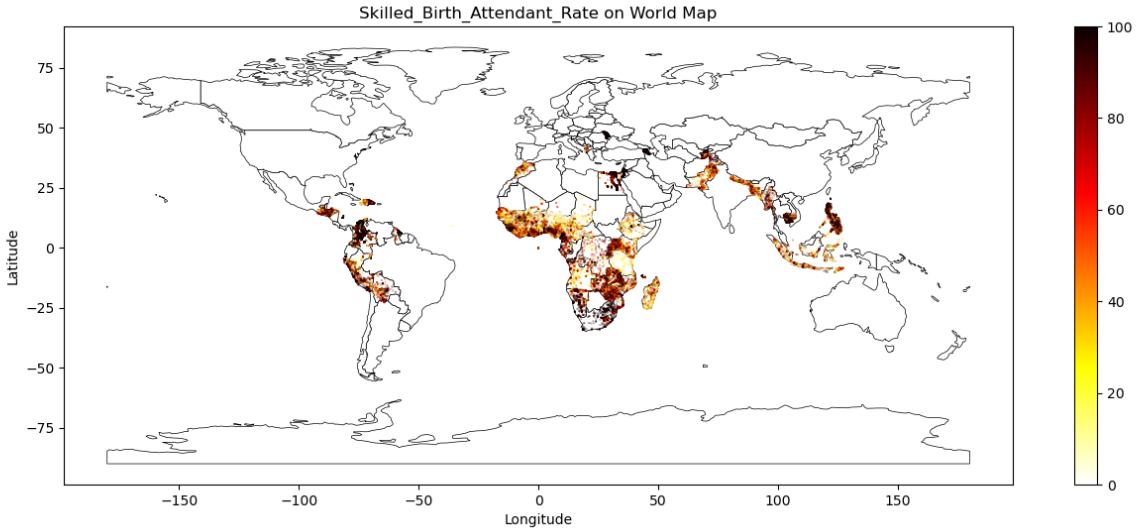


Figure 12: Skilled Birth Attendant Rate

A.2 Feature Selection

1. **Spearman Correlation**
   To calculate the Spearman correlation coefficient, we ranked the values of feature and target columns. We then applied the following formula:

$$1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2 - 1)} \tag{2}$$

   where d_i represents the difference between the ranks of each pair of corresponding data points and 'n' denotes the number of observations.
   A positive correlation (rho close to +1) indicates that higher ranks in one input feature tend

to be associated with higher ranks in a target column and vice versa. A correlation value close to 0 suggests a weak monotonic relationship between the variables. The significance of the correlation coefficient is assessed through appropriate statistical tests (e.g., p-value).

2. **Random Forest Feature Selection**
   For classification, the measure of impurity is either the Gini impurity or the information gain/entropy. For regression the measure of impurity is variance. Therefore, when training a tree, it is possible to compute how much each feature decreases the impurity. Each tree of the random forest can calculate the importance of a feature according to its ability to increase the pureness of the leaves. The more a feature decreases the impurity, the more important the feature is. This is done for each tree, then is averaged among all the trees and, finally, normalized to 1. So, the sum of the importance scores calculated by a Random Forest is 1.

A.3 Missing Labels

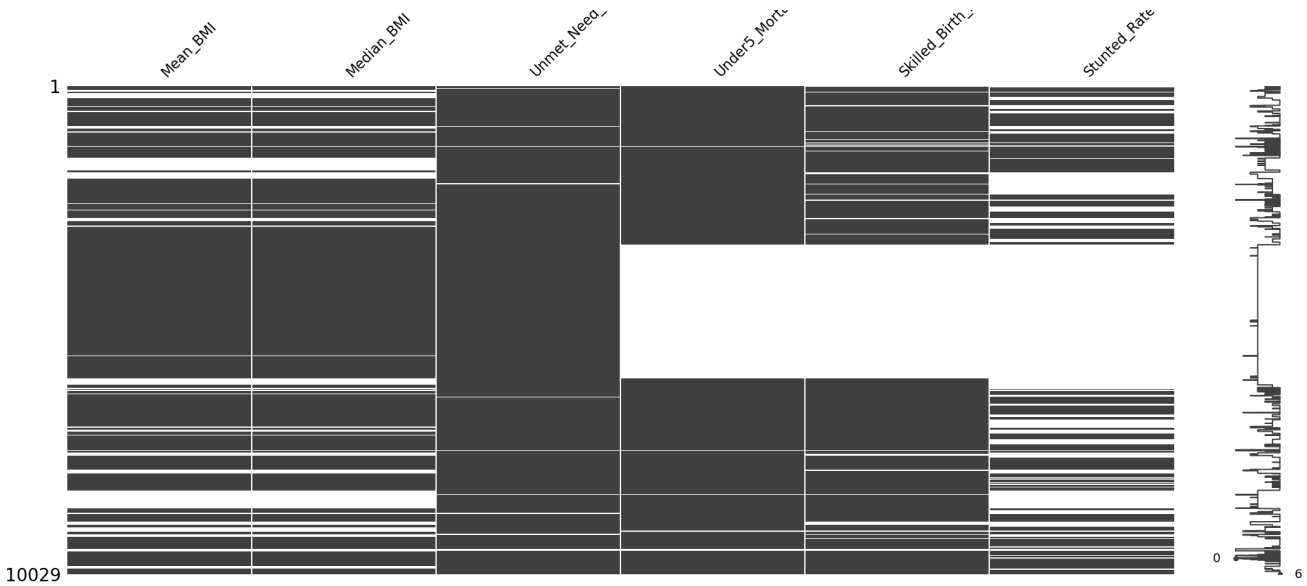


Figure 13: Missing values for each Label