**EARLY PREDICTION OF LOW BIRTH WEIGHT (LBW) CASES USING MACHINE LEARNING APPROACH**

|  |  |
| --- | --- |
| **CONTENT** | **PAGE NO** |
| **Abstract** |  |
| **1.INTRODUCTION** |  |
| 1.1 Motivation |  |
| 1.2 Problem Statement |  |
| 1.3 Objective of the Project |  |
| 1.4 Scope |  |
| 1.5 Project Introduction |  |
| **2.LTERATURE SURVEY** |  |
| 2.1 Related Work |  |
| **3. SYSTEM ANALYSIS** |  |
| 3.1 Existing System |  |
| 3.2 Disadvantages |  |
| 3.3 Proposed System |  |
| 3.4 Advantages |  |
| 3.5 work Flow of Proposed system |  |
| **4. REQUIREMENT ANALYSIS** |  |
| 4.1Function and non-functional requirements |  |
| 4.2 Hardware Requirements |  |
| 4.3 Software Requirements |  |
| 4.4 Architecture |  |
| **5. SYSTEM DESIGN** |  |
| 5.1 Introduction of Input design |  |
| 5.2 UML Diagram(class, use case, sequence, collaborative, deployment, activity, ER diagram and Component diagram) |  |
| 5.3 Data Flow Diagram |  |
| **6. IMPLEMENTATION** |  |
| 6.1 Modules |  |
| 6.2 Algorithms |  |
| 6.2.1 XGBoost |  |
| 6.2.2 Random Forest |  |
| 6.2.3 Decision Tree Classifier |  |
| 6.2.4 Support Vector classifier |  |
| 6.2.5 Stacking Algorithm |  |
| 6.3 Output Screens |  |
| 1. **PYTHON ENVIRONMENT** |  |
| 7.1 Software Installation |  |
| 7.2 About Python Language |  |
| 1. **SYSTEM STUDY AND TESTING** |  |
| 8.1 Feasibility study |  |
| 8.2 Types of test & Test Cases |  |
| **9. CONCLUSION** |  |
| **10. FUTURE ENHANCEMENT** |  |
| **11. REFERENCES** |  |

**ABSTRACT:**

Low Birth weight (LBW) acts as an indicator of sickness in newborn babies. LBW is closely associated with infant mortality as well as various health outcomes later in life. Various studies show strong correlation between maternal health during pregnancy and the child’s birth weight. This manuscript exploits machine learning techniques to gain useful information from health indicators of pregnant women for early detection of potential LBW cases. The forecasting problem has been reformulated as a classiﬁcation problem between LBW and NOT-LBW classes using supervised Machine learning for LBW detection as a binary machine classiﬁcation problem. Expectedly, the proposed model achieved better accuracy. Indian health care data was used to construct decision rules to be extrapolated to predictive health care in smart cities. A screening tool based on the decision model is developed to assist health care professionals in Obstetrics and Gynecology (OBG).**KEYWORDS:** Low Birth weight (LBW), Smart health informatics, Predictive analytics, Machine Learning (ML).

1. **INTRODUCTION**

**1.1 Motivation:**

Early prediction of low birth weight (LBW) cases using a machine learning approach is motivated by the urgent need to identify at-risk pregnancies in advance. By leveraging data-driven models, we aim to detect LBW cases at an early stage, enabling timely medical intervention and improved neonatal care. This proactive approach can significantly reduce the adverse health outcomes associated with LBW, enhancing the overall well-being of both mothers and infants.

**1.2 Problem Statement:**

The problem of early prediction of low birth weight (LBW) cases using a machine learning approach is a critical healthcare challenge. LBW infants face higher risks of health complications and mortality. Current methods for identifying these cases often lack accuracy and timeliness. Leveraging machine learning to develop a robust predictive model is essential to enable early intervention and improve neonatal outcomes. Addressing this problem can enhance maternal and infant healthcare by reducing LBW-related risks.

**1.3 Objective of the Project:**

The primary objective of this project is to develop a reliable machine learning-based system for the early prediction of low birth weight (LBW) cases. By leveraging data analysis and predictive modeling, we aim to identify pregnant individuals at risk of delivering LBW infants well in advance of birth. This system will enable timely medical interventions and targeted care strategies, ultimately improving neonatal health outcomes and reducing the associated risks for both mothers and infants.

**1.4 Scope:**

The scope of this project encompasses the development and implementation of a comprehensive machine learning solution for the early prediction of low birth weight (LBW) cases. It involves data collection, preprocessing, model training, and validation. The project will also explore various potential predictors and evaluate their impact on prediction accuracy. The ultimate goal is to create a practical and effective tool that can be integrated into existing healthcare systems to improve LBW risk assessment and maternal-infant care.

**1.5 Project Introduction:**

World Health Organization Maternal Health and Safe Motherhood Programme-1992, Low Birth Weight. It is expected to rise at the rate of 12% every year. Nearly 39% of power is used for cooling 45% for running the Information Technology (IT), infrastructure and 13% for lights. This level of consumption costs heavily to the businesses. LBW and prematurity remain a serious public health burden worldwide. Neonatal deaths account for a major fraction of deaths of children under the age of five, globally Children with LBW are at significantly higher risks of early childhood morbidity and mortality when compared with their counterparts with normal birth weights.

Low birth weight is the term used to refer to babies born with a weight less than 2500gm Low birth weight (LBW) has been identiﬁed as a major public health problem around the world. LBW includes both pre-term babies as well as fully grown babies who are very small in size as a consequence of intra uterine growth retardation. Birth weight is closely associated with neonatal and infant mortality, mortality rates being signiﬁcantly higher in LBW babies when compared to the normal birth weight (NBW) babies. This phenomenon is now of global concern in the view of serious short term and long term problems such as development disorders, neurosensory outcomes, health outcomes including Type 2 diabetes, cerebral stroke, hypertension and various other disorders that LBW babies are prone to. Studies in 2013 showed that out of the 22 million newborns about 16 percent were low birth weight cases globally. This is a major problem in developing countries, especially in India which contributes to about 30 percent of the global LBW cases.

Innumerable studies around the world indicate strong between maternal health and impact on birth weight of babies. Popular assumptions claim that LBW can be considerably reduced, with dedicated medical care during pregnancy. In our approach, the risk factors in pregnant women that can be easily assessed with basic methods are carefully examined throughout the gestation period and form the basis for predictions. Early detection can help in preventing the chances of LBW and also to put forward some recommendations under some intervention mechanisms. **LITERATURE SURVEY**

**2.1 Related Work:**

**[1] Kramer MS. Determinants of low birth weight: methodological assessment and meta-analysis. Bull World Health Organ. 1987; 65(5):663-737. PMID: 3322602; PMCID: PMC2491072.**

The existence and magnitude of a causal effect on birth weight, gestational age, and prematurity and intrauterine growth retardation were determined by a set of methodological standards. In developed countries, the most important factor was cigarette smoking, followed by nutrition and pre-pregnancy weight. In developing countries the major determinants were racial origin, nutrition, low pre-pregnancy weight, short maternal stature, and malaria. Pre-pregnancy weight, prior premature birth or miscarriage, diethylstilbestrol exposure and smoking were major determinants of gestational duration, but the majority of prematurity was unexplained in both developed and developing countries.

**[2] Vega J, Sáez G, Smith M, Agurto M, Morris NM. Factores de riesgo para bajo peso al nacer y retardo de crecimiento intrauterino en Santiago de Chile [Risk factors for low birth weight and intrauterine growth retardation in Santiago, Chile]. Rev Med Chil. 1993 Oct; 121(10):1210-9. Spanish. PMID: 8191127.**

An epidemiologic case-control study to ascertain the determinants of low birth weight was carried out in Santiago, Chile, from January to December 1989. The cases were defined as live births < 2500 g. The controls were live births > or = 2500 g of birth weight. All cases and a random sample (1:1) of controls were selected among 8,254 singleton births occurring at the El Salvador Hospital in the Eastern area of Santiago. These deliveries represented 50% of institutional deliveries in the area. Home deliveries (2%) and private hospital deliveries were not included in the study. Information was obtained from hospital medical records by six trained medical students. Some information could not be obtained from the hospital medical records. Thus the second step in data collection was the tracking of all the selected subjects to their referring neighborhood health centers.

**[3] Mavalankar DV, Trivedi CC, Gray RH. Maternal weight, height and risk of poor pregnancy outcome in Ahmedabad, India. Indian Pediatr. 1994 Oct; 31(10):1205-12. PMID: 7875780.**

This paper explores the relationships between maternal weight, height and poor pregnancy outcome using a data set from a case-control study of low birth weight (LBW) and perinatal mortality in Ahmedabad, India. Maternal height and weights were compared between mothers of 611 perinatal deaths, 644 preterm-LBW, and 1465 normal birth weight controls as well as 617 small-for-gestational age (SGA) and 1851 appropriate-for-gestational-age (AGA) births. Weight and height were much lower in this population compared to western standards. Low weight and height were associated with increased risk of perinatal death, prematurity and SGA. After adjusting for confounders, maternal weight remained significantly associated with poor pregnancy outcomes, whereas height was only weakly associated. Attributable risk estimates show that low weight is a much more important contributor to poor outcome than low height. Improvement in maternal nutritional status could lead to substantial improvement in birth outcome in this population

**[4] Bosetti C, Nieuwenhuijsen MJ, Gallus S, Cipriani S, La Vecchia C, Parazzini F. Ambient particulate matter and preterm birth or birth weight: a review of the literature. Arch Toxicol. 2010 Jun;84(6):447-60. doi: 10.1007/s00204-010-0514-z. Epub 2010 Feb 6. PMID: 20140425.**

To review epidemiologic evidence on maternal exposure to particulate matter and adverse pregnancy outcomes, we performed a MEDLINE search of the literature up to June 2009. We considered all original studies published in English including information on total suspended particles (TSP), respirable (PM(10)) or fine (PM(2.5)) particles and the risk of preterm birth, low birth weight (LBW) or very low birth weight (VLBW) and small for gestational age (SGA). We identified a total of 30 papers, including 13 with information on preterm birth, 17 on LBW or VLBW, and 4 on SGA. Eight studies on preterm birth, 11 studies on LBW/VLBW and two studies on SGA reported some increased risk (by about 10-20%) in relation to exposure to PM; no meaningful associations was found in the remaining studies. However, even in studies reporting some excess risk, this was inconsistent across exposure levels and pregnancy periods. Epidemiologic studies on maternal exposure to PM during pregnancy thus do not provide convincing evidence of an association with the risk of preterm birth and LBW/VLBW and SGA. The excess risks, if any, are small, and it is unclear whether they are causal, due to misclassification of the exposure or some sources of bias/residual confounding.

**3. SYSTEM ANALYSIS**

**3.1 Existing System**

In existing system, model used is Random Forest and Xgboost and Decision Tree to estimate whether the baby belongs to the Low Birth Weight or not belongs to the Low Birth. This model employs low accuracy and inaccurate results.

**3.2** **Disadvantages**

**1. Low accuracy:** The model's predictions do not align closely with the actual outcomes, resulting in a high rate of incorrect classifications and reduced trustworthiness.

**2. Expensive:** The implementation or maintenance of the solution incurs significant costs, potentially exceeding available resources and hindering widespread adoption.

**3. Low reliability:** The system lacks consistency in producing dependable results, making it unreliable for critical tasks or decision-making.

**4. Inaccurate:** The model's outputs deviate significantly from the ground truth, leading to incorrect conclusions and diminished utility in practical applications.

**PROPOSED SYSTEM**

In the proposed system, we employ a sophisticated approach that leverages supervised machine learning algorithms, specifically Stacking Algorithms and Support Vector Classifier (SVC), to enhance the prediction of low birth weight (LBW) cases in early pregnancy. Stacking enables the combination of diverse predictive models, while the SVC harnesses its classification capabilities to refine our LBW risk assessments. This comprehensive strategy harnesses the power of machine learning to create a robust and accurate framework for the early prediction of LBW cases, ultimately improving neonatal healthcare and maternal well-being.

**3.4 Advantages**

**1. High accuracy:** The system delivers exceptionally precise results, aligning closely with real-world outcomes, enhancing decision-making, and boosting confidence in its predictions.

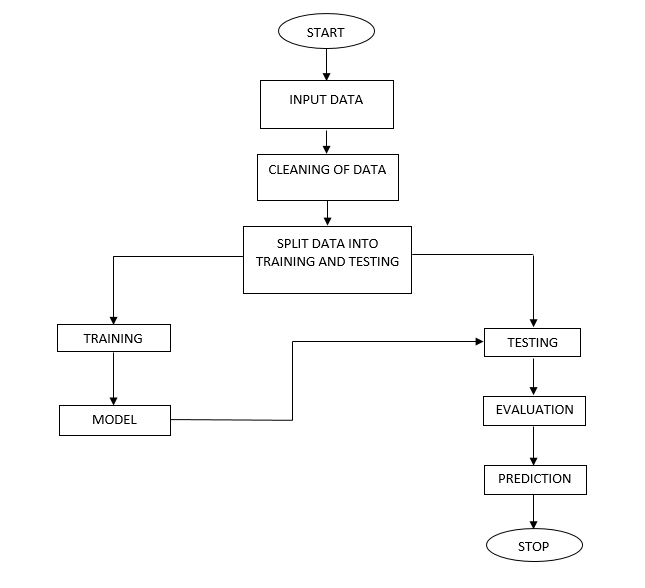
**2. Time-saving:** It significantly reduces the time required for tasks or processes, increasing overall efficiency and productivity.

**3. Does not require highly trained staff:** It can be operated effectively by individuals with minimal training, reducing the need for specialized expertise.

**4. High reliability:** The system consistently produces dependable results, ensuring its suitability for critical applications and maintaining user trust.

**5. Low complexities:** It features a straightforward, user-friendly design with minimal intricacies, enhancing usability and ease of implementation.

**3.5 work Flow of Proposed system**



**4. REQUIREMENT ANALYSIS**

**4.1 Functional and non-functional requirements**

Requirement’s analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

**Functional Requirements**: These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Examples of functional requirements:

1. Authentication of user whenever he/she logs into the system
2. System shutdown in case of a cyber-attack
3. A verification email is sent to user whenever he/she register for the first time on some software system.

**Non-functional requirements**: These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.  
They basically deal with issues like:

* Portability
* Security
* Maintainability
* Reliability
* Scalability
* Performance
* Reusability
* Flexibility

Examples of non-functional requirements:

1. Emails should be sent with a latency of no greater than 12 hours from such an activity.
2. The processing of each request should be done within 10 seconds
3. The site should load in 3 seconds whenever of simultaneous users are > 10000

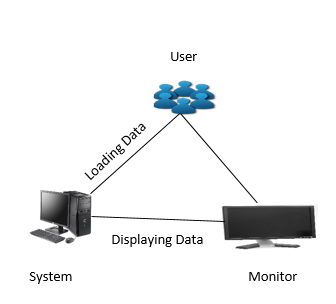
# **4.2Hardware Requirements**

# Processor : I3/Intel Processor

1. RAM : 4GB (min)
2. Hard Disk : 128 GB
3. Key Board : Standard Windows Keyboard
4. Mouse : Two or Three Button Mouse
5. Monitor : Any

**4.3Software Requirements:**

* Operating System : Windows 7+
* Server-side Script : Python 3.6+
* IDE : PyCharm.
* Libraries Used : Pandas, Numpy, Matplotlib, OS.
  1. **Architecture:**



**5. SYSTEM DESIGN**

## **Input Design:**

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

Therefore, the quality of system input determines the quality of system output. Well-designed input forms and screens have following properties −

It should serve specific purpose effectively such as storing, recording, and retrieving the information.

It ensures proper completion with accuracy.

It should be easy to fill and straightforward.

It should focus on user’s attention, consistency, and simplicity.

All these objectives are obtained using the knowledge of basic design principles regarding −

What are the inputs needed for the system?

How end users respond to different elements of forms and screens.

### **Objectives for Input Design:**

The objectives of input design are −

To design data entry and input procedures

To reduce input volume

To design source documents for data capture or devise other data capture methods

To design input data records, data entry screens, user interface screens, etc.

To use validation checks and develop effective input controls.

**Output Design:**

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

### **Objectives of Output Design:**

The objectives of input design are:

To develop output design that serves the intended purpose and eliminates the production of unwanted output.

To develop the output design that meets the end user’s requirements.

To deliver the appropriate quantity of output.

To form the output in appropriate format and direct it to the right person.

To make the output available on time for making good decisions

**5.2 UML Diagrams:**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

**USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



**CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



**SEQUENCE DIAGRAM:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



**COLLABORATION DIAGRAM:**

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



**DEPLOYMENT DIAGRAM**

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware used to deploy the application.



**ACTIVITY DIAGRAM:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



**COMPONENT DIAGRAM**

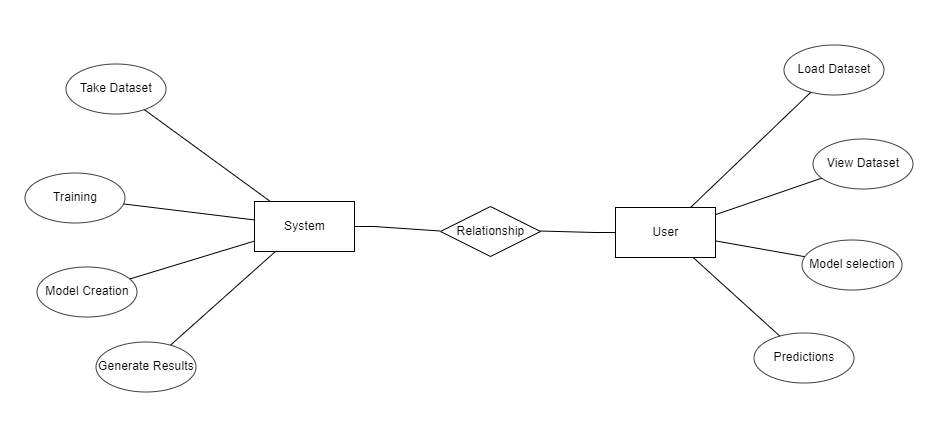
A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical **c**omponents in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.



**ER DIAGRAM:**

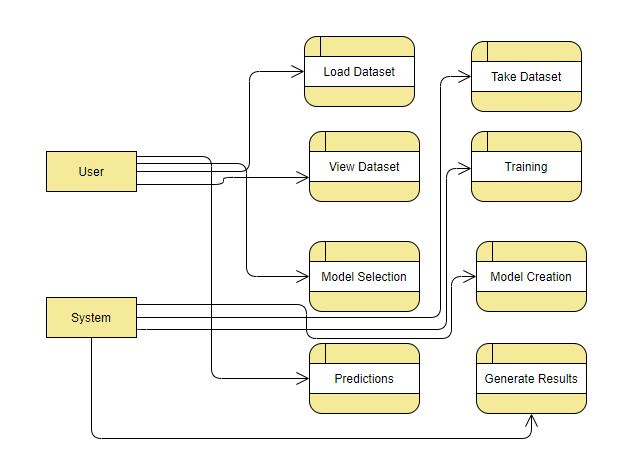
An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

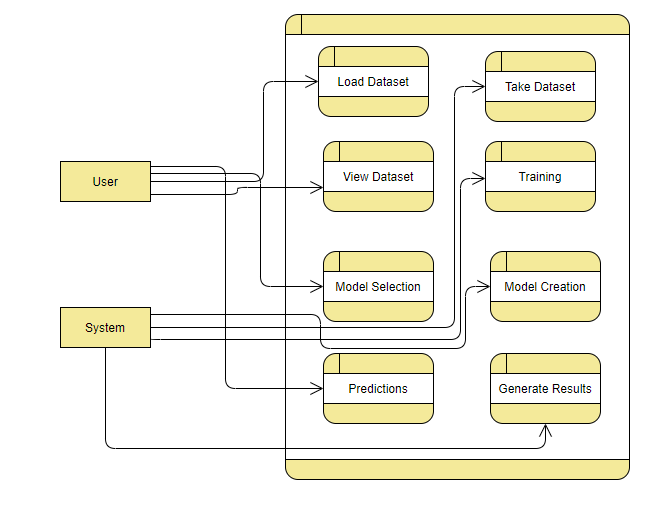
An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let’s have a look at a simple ER diagram to understand this concept.



**DFD DIAGRAM:**

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.





**IMPLEMENTATION AND RESULTS**

**1. System:**

**1.1 Takes Dataset:**This module allows users to upload their dataset, providing the necessary input data for the prediction process.

**1.2 Store Dataset:**Once the dataset is uploaded, the system stores it for subsequent use, ensuring data availability throughout the process.

**1.3 Model Selection:**Users can choose from a selection of machine learning models available within the system. This choice is critical as it determines the algorithm that will be used for prediction.

**1.4 Model Predictions:** After model selection, the system utilizes the chosen model to make predictions based on the user-provided data, generating valuable insights related to LBW risk.

**2. User:**

**2.1 Load Dataset:**Users have the flexibility to load the specific dataset they intend to work with, enabling customization and compatibility with their data.

**2.2 View Dataset:**This module offers users the capability to visually inspect the uploaded dataset, facilitating data exploration and understanding.

**2.3 Select Model:** Users can select a machine learning model from those made available by the system. The choice may depend on the specific dataset and the desired level of prediction accuracy.

**2.4 Predictions:** In this module, users can input random or specific values related to maternal and pregnancy factors to obtain predictions about the likelihood of LBW, allowing for scenario analysis and proactive decision-making.

**6.2Algorithms**

**6.2.1 XGBoost:**

XGBoost is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks. However, when it comes to small-to-medium structured/tabular data, decision tree based algorithms are considered best-in-class right now.

Bagging: Now imagine instead of a single interviewer, now there is an interview panel where each interviewer has a vote. Bagging or bootstrap aggregating involves combining inputs from all interviewers for the final decision through a democratic voting process.

XGBoost and Gradient Boosting Machines (GBMs) are both ensemble tree methods that apply the principle of boosting weak learners (CARTs generally) using the gradient descent architecture. However, XGBoost improves upon the base GBM framework through systems optimization and algorithmic enhancements.

**6.2.2 Random Forest:**

First, Random Forest algorithm is a supervised classification algorithm. We can see it from its name, which is to create a forest by some way and make it random. There is a direct relationship between the number of trees in the forest and the results it can get: the larger the number of trees, the more accurate the result. But one thing to note is that creating the forest is not the same as constructing the decision with information gain or gain index approach.

The author gives four advantages to illustrate why we use Random Forest algorithm. The one mentioned repeatedly by the author is that it can be used for both classification and regression tasks. Overfitting is one critical problem that may make the results worse, but for Random Forest algorithm, if there are enough trees in the forest, the classifier won’t overfit the model. The third advantage is the classifier of Random Forest can handle missing values, and the last advantage is that the Random Forest classifier can be modeled for categorical values.

There are two stages in Random Forest algorithm, one is random forest creation, the other is to make a prediction from the random forest classifier created in the first stage.

**STEPS:**

1. Randomly select “K” features from total “m” features where k << m
2. Among the “K” features, calculate the node “d” using the best split point
3. Split the node into daughter nodes using the best split
4. Repeat the a to c steps until “l” number of nodes has been reached
5. Build forest by repeating steps a to d for “n” number times to create “n” number of trees

**6.2.3 Decision Trees:**

A tree has many analogies in real life, and turns out that it has influenced a wide area of machine learning, covering both classification and regression. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. As the name goes, it uses a tree-like model of decisions. Though a commonly used tool in data mining for deriving a strategy to reach a particular goal.

A decision tree is drawn upside down with its root at the top. In the image on the left, the bold text in black represents a condition/internal node, based on which the tree splits into branches/ edges. The end of the branch that doesn’t split anymore is the decision/leaf, in this case, whether the passenger died or survived, represented as red and green text respectively.

Although, a real dataset will have a lot more features and this will just be a branch in a much bigger tree, but you can’t ignore the simplicity of this algorithm. The feature importance is clear and relations can be viewed easily. This methodology is more commonly known as learning decision tree from data and above tree is called Classification tree as the target is to classify passenger as survived or died. Regression trees are represented in the same manner, just they predict continuous values like price of a house. In general, Decision Tree algorithms are referred to as CART or Classification and Regression Trees.

So, what is actually going on in the background? Growing a tree involves deciding on which features to choose and what conditions to use for splitting, along with knowing when to stop. As a tree generally grows arbitrarily, you will need to trim it down for it to look beautiful. Let’s start with a common technique used for splitting.

**6.2.4 Support Vector Machine:**

A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for each category, they’re able to categorize new text.

So you’re working on a text classification problem. You’re refining your training data, and maybe you’ve even tried stuff out using Naive Bayes. But now you’re feeling confident in your dataset, and want to take it one step further. Enter Support Vector Machines (SVM): a fast and dependable classification algorithm that performs very well with a limited amount of data to analyze.

Perhaps you have dug a bit deeper, and ran into terms like linearly separable, kernel trick and kernel functions. But fear not! The idea behind the SVM algorithm is simple, and applying it to natural language classification doesn’t require most of the complicated stuff.

**Steps for implementation:**

* Import the dataset.
* Explore the data to figure out what they look like.
* Pre-process the data.
* Split the data into attributes and labels.
* Divide the data into training and testing sets.
* Train the SVM algorithm.
* Make some predictions

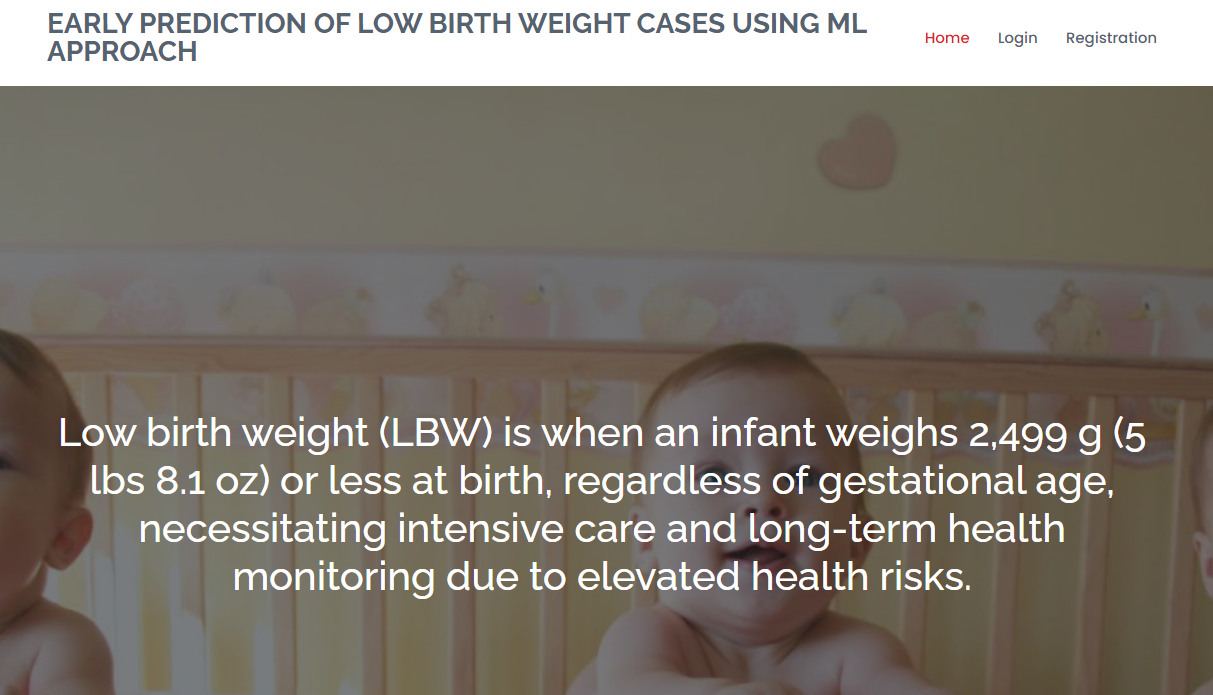
**6.2.5 Stacking Algorithm:**

Stacking is an ensemble learning technique that combines the predictions of multiple machine learning models to improve overall predictive performance. It aims to harness the strengths of different base models and create a meta-model, or "stacker," that learns how to best combine their outputs. The process typically involves two or more levels of models. In the first level, various base models are trained on the same dataset. These base models can be diverse in terms of algorithms, such as decision trees, support vector machines, or neural networks. In the second level, a meta-model, often a simple linear regression or another machine learning algorithm, is trained on the predictions made by the base models in the first level. Stacking leverages the complementary strengths of base models to make more accurate predictions and is known for its ability to handle complex relationships in data. It can be particularly effective in tasks where individual models may struggle, ultimately enhancing the predictive power of machine learning systems. However, it requires careful tuning and may be computationally intensive due to its multi-level nature.

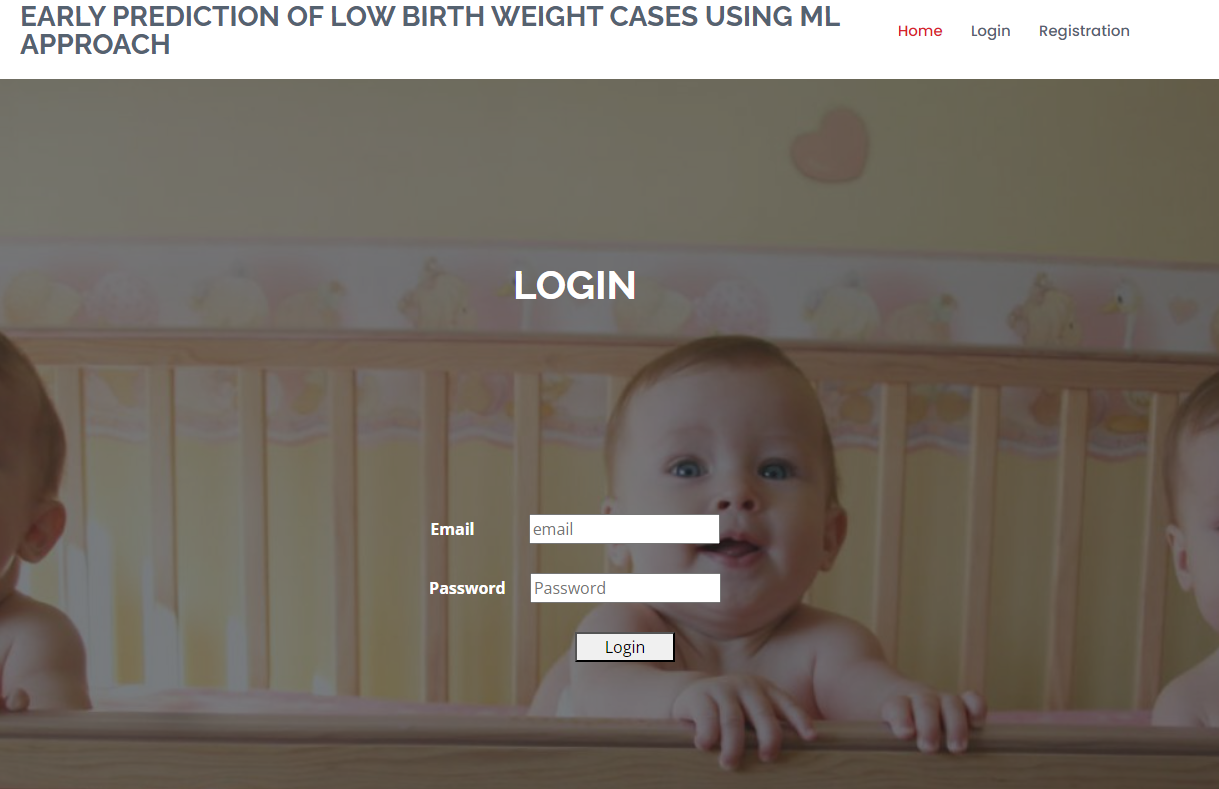
**6.3Output Screens**

**HOME PAGE:**

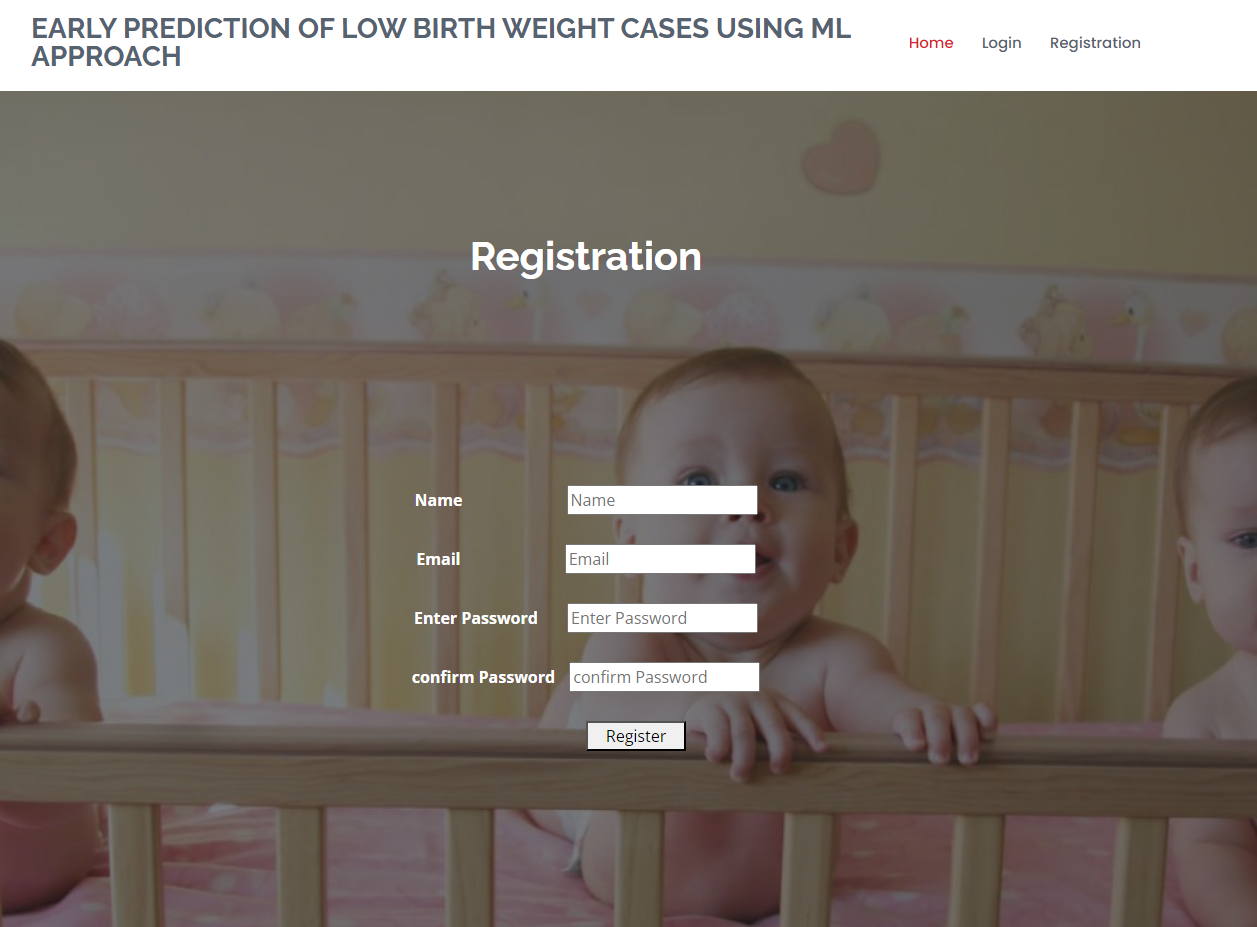
This the homepage of lower birth weight prediction web application.



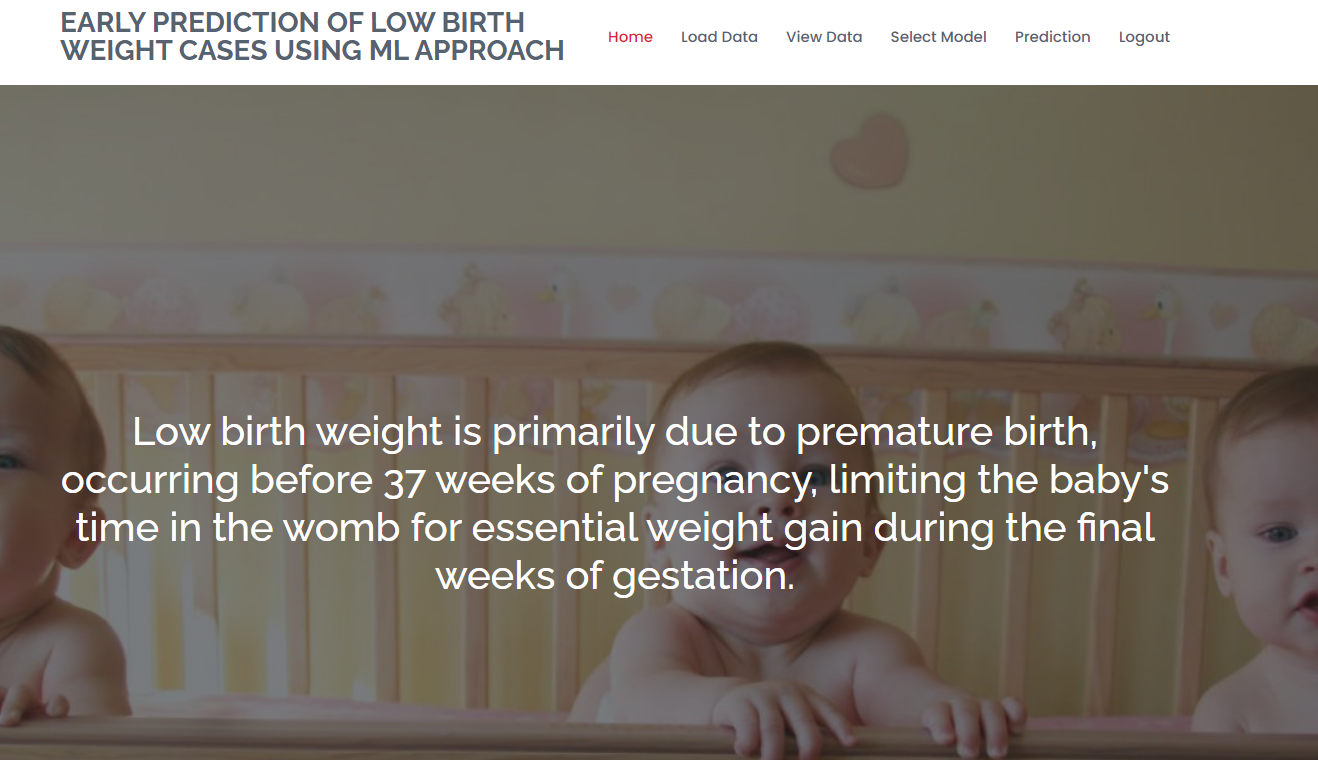
**Login page:**



**Registration:**

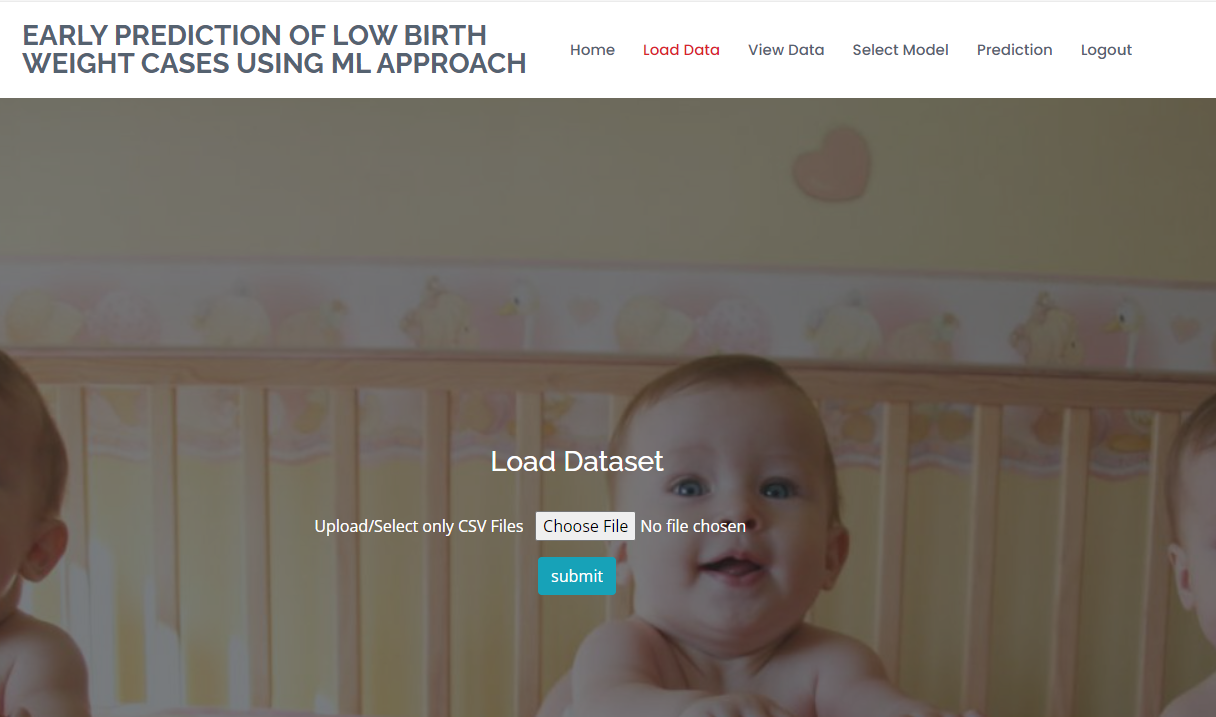


**User Home Page:**



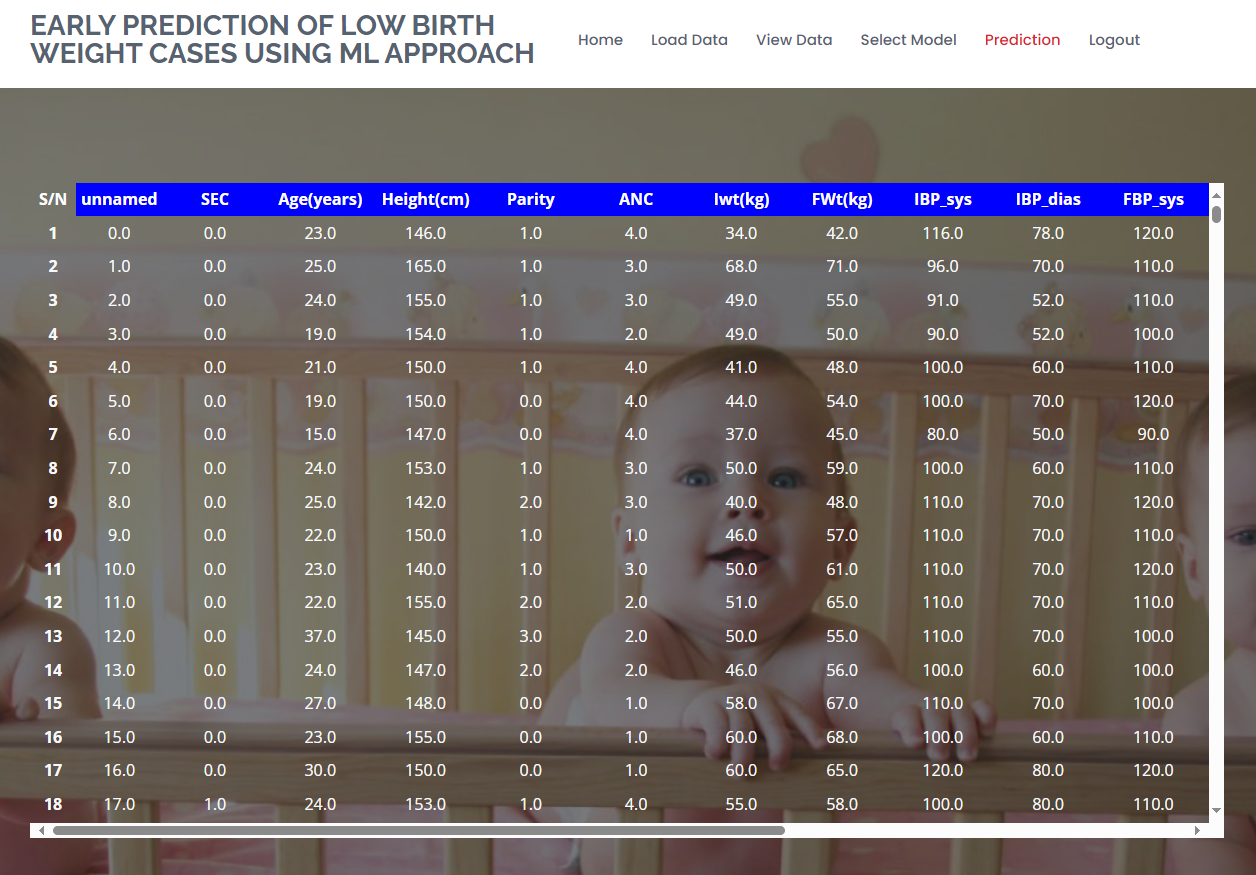
**DATA LOADING PAGE:**

In this page user have to upload the dataset.



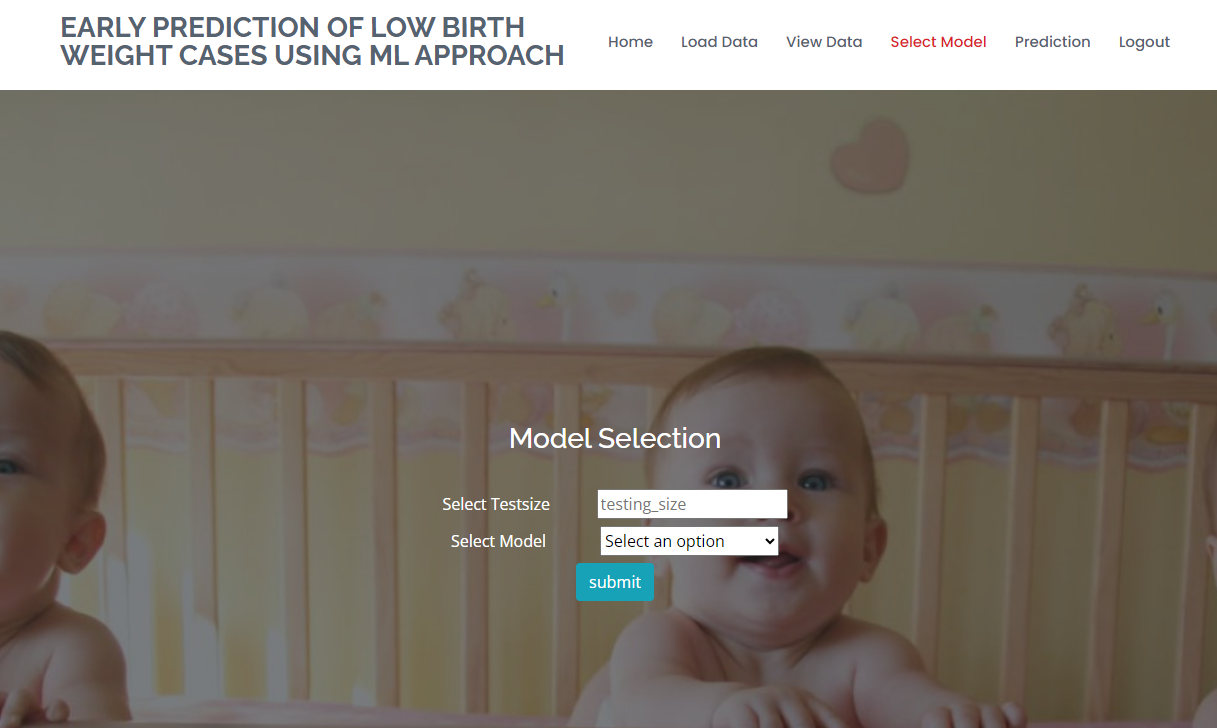
**DATA VIEWING PAGE:**

User can view the data in this page after sending dataset as input.



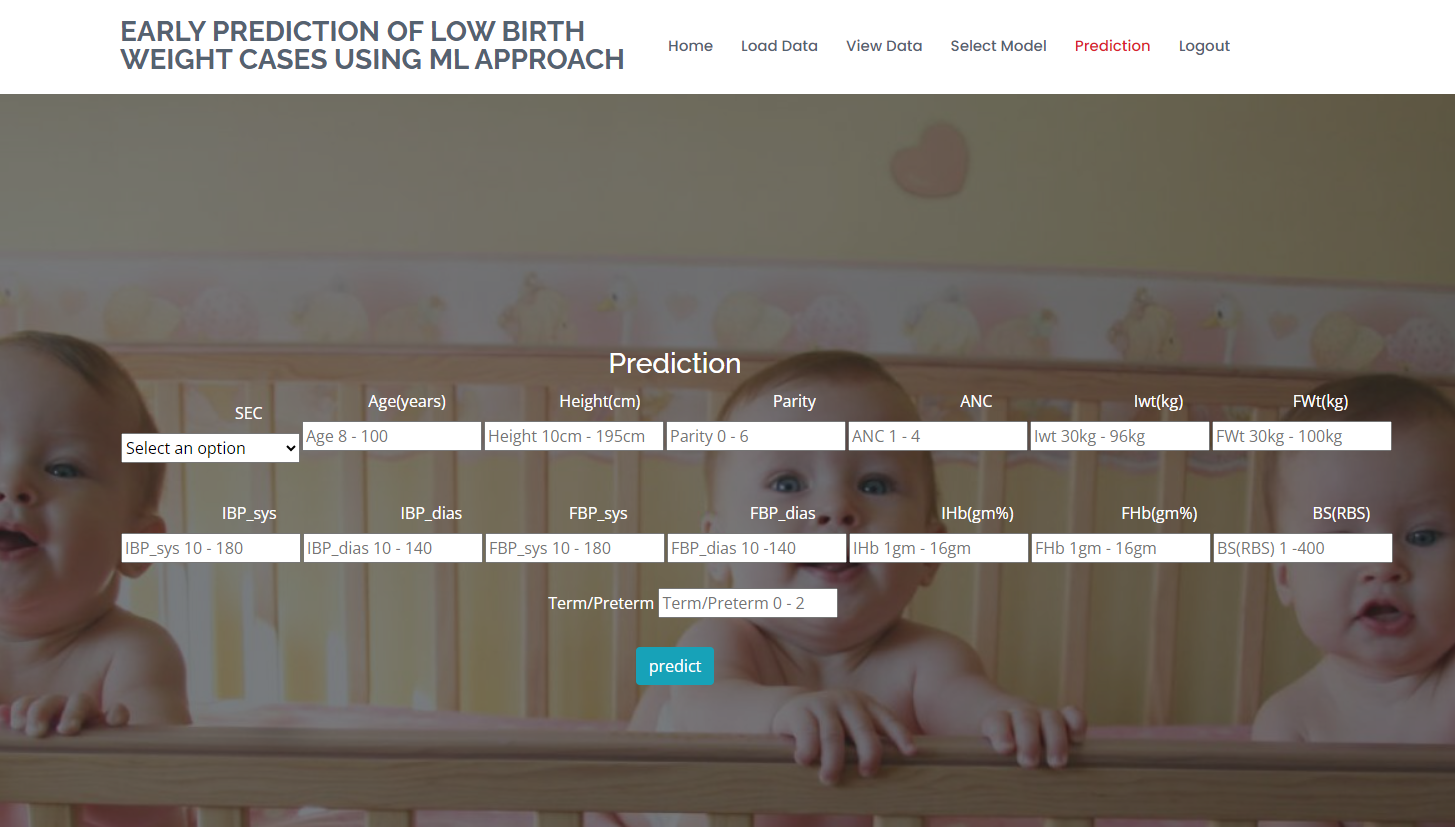
**MODEL SELECTION:**

This page is model selection page. User can provide the model and text size.



**PREDICTION PAGE:**

User can view the prediction page after providing the input dataset



**8. SYSTEM STUDY AND TESTING**

**8.1 Feasibility Study**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### **TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**SYSTEM TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**TYPES OF TESTS**

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**SYSTEM TEST**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**6.1 Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

# 6.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**6.3 Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**9.CONCLUSION**

In this application, we have successfully created a ML models to estimate whether the baby belongs to the Low Birth Weight or not belongs to the Low Birth. This is developed in a user friendly environment using Flask via Python programming. We noticed that out of XGBoost Classifier, Random Forest Classifier, Decision Tree Classifier, Support Vector Classifier Decision Tree Classifier and stacking classifier performs well with better accuracy.

**10. FUTURE ENHANCEMENT**

Future enhancements for the early prediction of low birth weight (LBW) cases using machine learning can encompass several avenues of improvement. Firstly, the integration of more comprehensive and diverse data sources, including genetic, environmental, and lifestyle factors, can refine predictive models for greater accuracy.Additionally, the development of real-time monitoring systems, using wearable devices and continuous data streams, could enable dynamic risk assessment throughout pregnancy. This could facilitate timely interventions and personalized care plans.Moreover, leveraging advanced deep learning techniques, such as neural networks and recurrent neural networks, may capture complex temporal patterns in maternal health data more effectively, further enhancing prediction accuracy.Lastly, collaboration with healthcare professionals and institutions can ensure the seamless integration of machine learning models into clinical practice, fostering a holistic approach to LBW risk mitigation. These future enhancements collectively hold the potential to revolutionize LBW prediction, leading to improved maternal and neonatal healthcare outcomes.

**11.REFERANCES**

[1] World Health Organization-1992, International statistical classiﬁcation of diseases and related health problems,Tenth revision, Geneva, World Health Organization.

[2] Kramer MS. Determinants of low birth weight: methodological assessment and meta-analysis. Bull World Health Organ. 1987; 65(5):663-737. PMID: 3322602; PMCID: PMC2491072.

[3] Vega J, Sáez G, Smith M, Agurto M, Morris NM. Factores de riesgo para bajo peso al nacer y retardo de crecimiento intrauterino en Santiago de Chile [Risk factors for low birth weight and intrauterine growth retardation in Santiago, Chile]. Rev Med Chil. 1993 Oct; 121(10):1210-9. Spanish. PMID: 8191127.

[4] Mavalankar DV, Trivedi CC, Gray RH. Maternal weight, height and risk of poor pregnancy outcome in Ahmedabad, India. Indian Pediatr. 1994 Oct; 31(10):1205-12. PMID: 7875780.

[5] Bosetti C, Nieuwenhuijsen MJ, Gallus S, Cipriani S, La Vecchia C, Parazzini F. Ambient particulate matter and preterm birth or birth weight: a review of the literature. Arch Toxicol. 2010 Jun; 84(6):447-60. Doi: 10.1007/s00204-010-0514-z. Epub 2010 Feb 6. PMID: 20140425.

[6] United nations Children’s Fund and World Health Organization2004, Low Birth Weight: Country, regional and global estimates, New York, UNICEF.

[7] J.S. Deshmukh, D.D. Motghare, S.P. Zodpey and S.K. Wadhva1998, Low Birth Weight And Associated Maternal Factors in an Urban Area, Indian Pediatrics,Volume 35, Page 33-36.

[8] J.S. Deshmukh, D.D. Motghare, S.P. Zodpey and S.K. Wadhva1998, Low Birth Weight And Associated Maternal Factors in an Urban Area, Indian Pediatrics,Volume 35, Page 33-36.

[9] Aparajita Dasgupta, Rivu Basu, “Determinants of low birth weight in a Blowk of Hooghly, West Bengal: A multivariate analysis,” International Journal of Biological & Medical Research, 2(4), 2011, pp.838-842.

[10] Bellazzi R, Zupan B., “Towards knowledge-based gene expression data mining”, J Biomed Inform. 2007; 40(6), pp.787-802