

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

*A Project Report submitted in partial fulfillment of the requirements for
the award of the degree of*

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

Submitted By

1. KARRI HEMA HARSHITHA (19B01A0567)
2. KOMALI ARTHI (19B01A0574)
3. MUDUNURI TULASI LAKSHMI (19B01A05B4)
4. NANDIKAM LAKSHMI SAI AKHILA (19B01A05C6)
5. NARIBOINA VANAJA (20B05A0511)

Under the esteemed guidance of
Mrs. K.Ratna Kumari (M.Tech)
Assistant Professor



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN(A)

(Approved by AICTE, Accredited by NBA & NAAC, Affiliated to JNTU Kakinada)

BHIMAVARAM – 534202

2022 – 2023

**SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN
(AUTONOMOUS)**

**(Approved by AICTE, Accredited by NBA & NAAC, Affiliated to JNTU Kakinada)
BHIMAVARAM – 534202**

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



CERTIFICATE

*This is to certify that the Major Project entitled "**EARLY PREDICTION OF LOW BIRTH WEIGHT(LBW) CASES USING MACHINE LEARNING**", is being submitted by **K.HEMA HARSHITHA, K.ARTHI, M.TULASI LAKSHMI, N.L.S.AKHILA, N.VANAJA** bearing the **Regd. No. 19B01A0567, 19B01A0574, 19B01A05B4, 19B01A05C6, 20B05A0511** in partial fulfillment of the requirements for the award of the degree of "**Bachelor of Technology in Computer Science & Engineering**" is a record of bonafide work carried out by them under my guidance and supervision during the academic year 2022–2023 and it has been found worthy of acceptance according to the requirements of the university.*

Internal Guide

Head of the Department

External Examiner

ACKNOWLEDGMENT

Behind every achievement there lies an unfathomable sea of graduates to those who activated it, without whom it would ever come into existence. To them we word of gratitude imprinted within us.

We wish to place our deep sense of gratitude to **Sri. K. V. Vishnu Raju, Chairman of SVES**, for providing excellent infrastructure to carry out our project .

We wish to express our sincere thanks to **Dr. G. Srinivasa Rao, Principal of SVECW**, for being a source of inspirational constant encouragement.

We wish to express our sincere thanks to **Dr. P. Srinivasa Raju, Vice-Principal of SVECW**, for being a source of inspirational constant encouragement.

On the successful completion of our project report, we convey our sincere thanks to **Dr. Kiran Sree, Head of the Department of Computer Science and Engineering**, for his valuable pieces of advice in completing this project successfully.

We are really thankful to our internal guide **Mrs.K.Ratna Kumari (M.Tech),Asst Professor**,who encouraged us a lot in the completion of the project work.

Project Associates

K.Hema Harshitha (19B01A0567)

K.Arthi (19B01A0574)

M.Tulasi Lakshmi (19B01A5B4)

N.L.S.Akhila (19B01A05C6)

N.Vanaja (20B05A0511)

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 a 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 healthcare data was used to construct decision rules to be extrapolated to predictive healthcare in smart cities. A screening tool based on the decision model is developed to assist healthcare professionals in Obstetrics and Gynecology (OBG).

CONTENTS

		Page.No
1	Introduction	1
	1.1 Common Problems of LBW	2
	1.2 Motivation For the Work	2
	1.3 Problem Statement	3
2	System Analysis	4
	2.1 Existing System	4
	2.2 Proposed System	5
	2.3 Feasibility Study	7
3	System Requirements Specification	9
	3.1 Software Requirements	11
	3.2 Hardware Requirements	11
4	System Design	12
	4.1 Introduction	14
	4.1.1 System Model	15
	4.2 UML Diagrams	16
5	Methodology	25
6	System Implementation	27
	6.1 Introduction	28
	6.2 Project Modules	28
	6.3 Screens	34
7	System Testing	38
	7.1 Introduction	39
	7.2 Testing Methods	41
	7.3 Testing Types	42
	7.4 Test Cases	44
8	Conclusion	45
9	Bibiliography	47
10	Appendix	50
	10.1 Software Installation for this Project	51
	10.2 Introduction to Python	54
	10.3 Some Python Libraries	54
	10.4 Why Python for Machine Learning?	57
	10.5 Introduction to FLASK Framework	58
	10.6 Machine Learning Algorithms	59

LIST OF FIGURES

		Page No.
1	Proposed System	7
2	System Model	15
3	Use Case Diagram	17
4	Class Diagram	18
5	Sequence Diagram	19
6	Collaboration Diagram	20
7	Activity Diagram	21
8	Deployment Diagram	22
9	Component Diagram	22
10	ER Diagram	23
11	DFD Diagram	24
12	Python Installation	51
13	Downloading PyCharm	52
14	Creating Project in PyCharm	53
15	Installing Packages	53

LIST OF SCREENS

		Page No.
1	Home Page	34
2	Data Uploading Page	34
3	Data Viewing Page	35
4	Model Selection Page	35
5	Prediction Page	36
6	Low-Birth-Weight Case	37
7	Not Low-Birth-Weight-Case	37

INTRODUCTION

1. 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 intrauterine growth retardation. Birth weight is closely associated with neonatal and infant mortality, mortality rates being significantly higher in LBW babies when compared to 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 developmental disorders, neurosensory outcomes, and 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 a strong between maternal health and the impact on the 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 put forward some recommendations under some intervention mechanisms.

1.1. COMMON PROBLEMS OF LBW:

Some of the common problems of low-birth-weight babies include:

- Low oxygen levels at birth
- Trouble staying warm
- Trouble feeding and gaining weight
- Infection
- Breathing problems and immature lungs (infant respiratory distress syndrome)
- Nervous system problems, such as bleeding inside the brain (intraventricular hemorrhage)
- Digestive problems, such as serious inflammation of the intestines (necrotizing enterocolitis)
- Sudden infant death syndrome (SIDS)

Babies with very low birth weight are at risk for long-term complications and disability. Long-term complications may include:

- Cerebral palsy
- Blindness
- Deafness
- Developmental delay

1.2. MOTIVATION FOR THE WORK:

Low birth weight is a term used to describe babies who are born weighing less than 5 pounds, 8 ounces (2,500 grams). An average newborn usually weighs about 8 pounds. A low-birth-weight baby may be healthy even though he or she is small. But a low-birth-weight baby can also have many serious health problems.

Another cause of low birth weight is a condition called intrauterine growth restriction (IUGR). This occurs when a baby does not grow well during pregnancy. It may be because of problems with the placenta, the mother's health, or the baby's health. Babies can have IUGR and be:

Full term. That means born from 37 to 41 weeks of pregnancy. These babies may be physically mature, but small.

Premature. These babies are both very small and physically immature.

So, Early prediction of low birth weight is an important foundation in reducing the ill effects to infants. This model uses classification problems to classify whether the baby born is either low birth weight or not low birth weight.

1.3. PROBLEM STATEMENT:

- Low birth weight (LBW) has been identified as a major public health problem around the world.
- By this application we predict LBW early so that special care and facilities are to be given to them when they are born so that their health condition will be improved.
- The technique we are using for Low-Birth-Weight case prediction is the best among the supervised Machine Learning algorithms like Decision Trees Random Forest, Support Vector Machine, and XGBoost.
- So, this Accurate Prediction System using machine learning mainly deals with the results of how a baby is born i.e. either low birth weight or not.

SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

2.1. EXISTING SYSTEM:

- In the existing system, the model used is Logistic Regression 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.
- Another way is fundal height:
- To check fundal height, your healthcare provider measures from the top of your pubic bone to the top of your uterus (fundus).
- Fundal height is measured in centimeters (cm). It is about the same as the number of weeks of pregnancy after the 20th week. For example, at 24 weeks gestation, your fundal height should be close to 24 cm.
- If the fundal height is less than expected, it may mean the baby is not growing well.

DISADVANTAGES:

- Low accuracy.
- Expensive.
- Low reliability.
- Inaccurate.

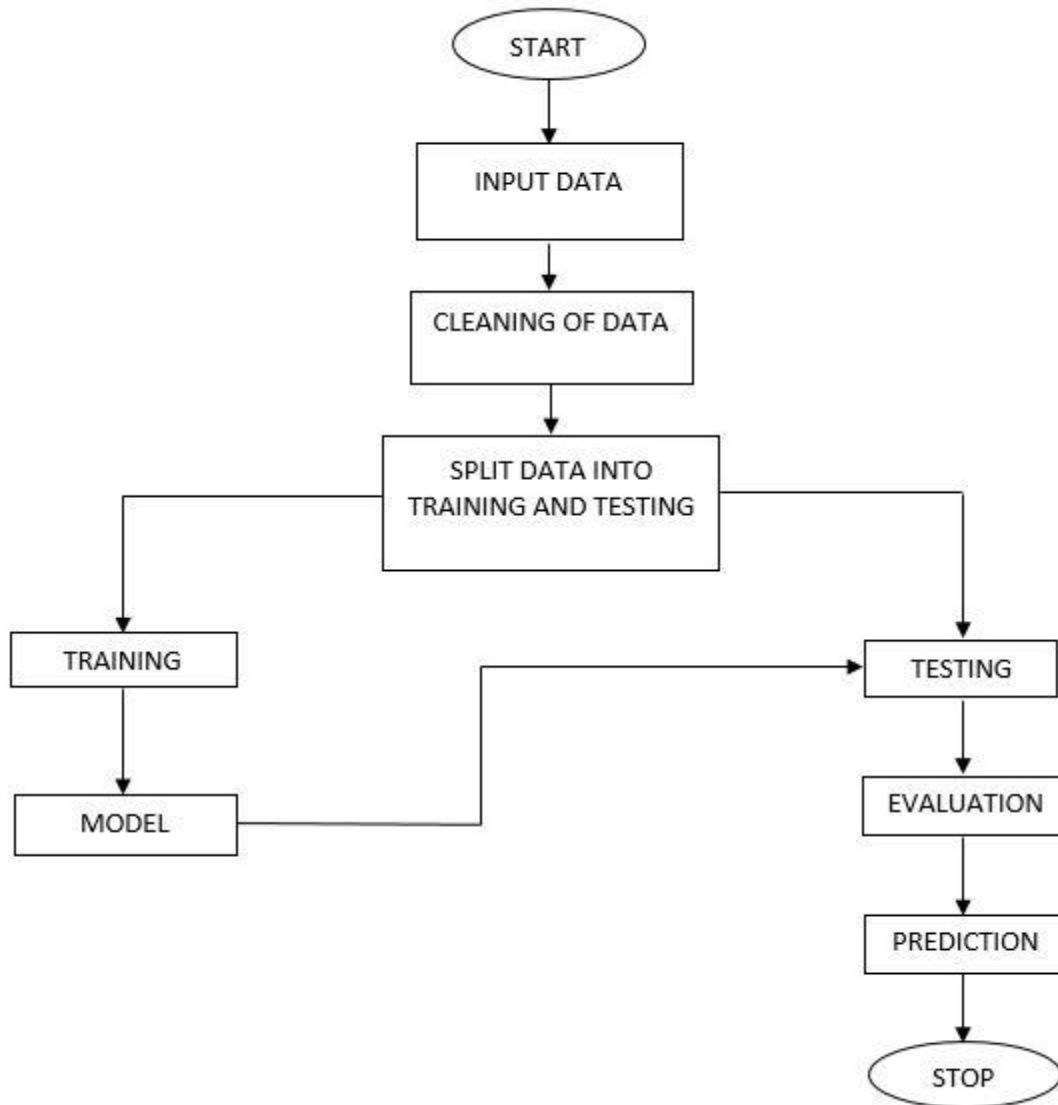
2.2. PROPOSED SYSTEM:

In proposed system, we implement supervised machine learning algorithms like XGBoost Classifier, Random Forest Classifier and Support Vector Classifier and Decision Tree Classifier for prediction of low Birth Weight babies.

ADVANTAGES:

- High accuracy.
- Time Saving.
- Does not require highly trained staff.
- High reliability.
- Low complexities.

FLOW DIAGRAM OF PROPOSED SYSTEM:



PROPOSED SYSTEM

2.3. FEASIBILITY STUDY:

The feasibility of the project is analyzed in this phase and the 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.

Generally, the feasibility study is used for determining the resource cost, benefits, and whether the proposed system is feasible with respect to the organization. The proposed system feasibility could be as follows.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

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 for 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.

Economic Feasibility:

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system is 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.

Social Feasibility:

The aspect of the 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 REQUIREMENTS SPECIFICATION

3. SYSTEM REQUIREMENTS SPECIFICATION

An SRS is a document that sets out what the client expects and what is expected of the software system which is being developed. It is a mutual agreement and insurance policy between the client and developer and is a vital part of the Software Development Lifecycle.

It is a description of a software system to be developed. It lays out functional and non-functional requirements and may include a set of use cases that describe the user interaction that the software must provide. Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers (in market-driven projects, these roles may be played by the marketing and development divisions) on what the software product is to do well as what it is not expected to do.

Software requirements specification permits a rigorous assessment of requirements before design can begin and reduces later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules. Used appropriately, software requirements specifications can help prevent software project failure.

The software requirements specification document enlists enough necessary requirements that are required for the project development. To derive the requirements, we need to have a clear and thorough understanding of the products to be developed or being developed. This is achieved and refined with detailed and continuous communications with the project team and customer till the completion of the software.

Requirement analysis can be a long and tiring process during which many delicate psychological skills are involved. Large systems may confront analysts with hundreds or thousands of system requirements. New systems change the environment and relationships between people, so it is important to identify all the stakeholders, consider all their needs to ensure they understand the implications of the new system. Analysis can employ several techniques to elicit the requirements of the customer.

3.1. SOFTWARE REQUIREMENTS

Software Requirements defines as functions that are required within a system that supports our project.

Operating System : Windows

It is compatible with various types of software and hardware.

Technology : Python

It is stable, flexible and allows using of various tools which makes the jobs easier.

Libraries Used : Pandas, Numpy, pymysql, random

A Python library is simply a collection of codes or modules of codes that we can use in a program for specific operations. We use libraries so that we don't need to write the code again in our program that is already available.

IDE : Any python IDE

Integrated Development Environment (IDE) integrates several tools specifically designed for software development. We used PyCharm.

Framework : Flask

It is an open-source framework for backend web applications in python.

3.2. HARDWARE REQUIREMENTS

Hardware Requirements defines as the physical resources which support the system to run an application smoothly.

Hardware : I3 and above

Our prediction system will work on i3 and above processor

RAM : 4GB and above

RAM stands for random access memory. It is a short-term memory used to handle all active tasks and apps.

Hard Disk : 128GB and above

Our prediction system will work on 128GB and above the hard disk.

SYSTEM DESIGN

4. SYSTEM DESIGN

4.1. INTRODUCTION

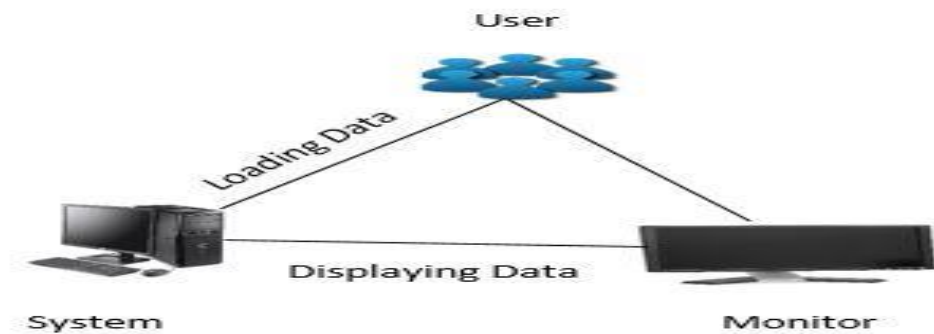
System design is the process of designing the elements of a system such as the architecture, modules and components, the different interfaces of those components and the data that goes through that system.

System Analysis is the process that decomposes a system into its component pieces for the purpose of defining how well those components interact to accomplish the set requirements. The purpose of the System Design process is to provide sufficient detailed data and information about the system and its system elements to enable the implementation consistent with architectural entities as defined in models and views of the system architecture.

The purpose of the design phase is to plan a solution of the problem specified by the requirement document. This phase is the first step in moving from problem domain to the solution domain. The design of a system is perhaps the most critical factor affecting the quality of the software, and has a major impact on the later phases, particularly testing and maintenance. The output of this phase is the design document. This document is similar to a blueprint or plan for the solution, and is used later during implementation, testing and maintenance.

A design methodology is a systematic approach to creating a design by application of set of techniques and guidelines. Most methodologies focus on system design. The two basic principles used in any design methodology are problem partitioning and abstraction. A large system cannot be handled as a whole, and so for design it's partitioned into smaller systems. Abstraction is a concept related to problem partitioning. When partitioning is used during design, the design activity focuses on one part of the system at a time. Since the part being designed interacts with other parts of the system, a clear understanding of the interaction is essential for properly designing the part.

4.1.1. SYSTEM MODEL:



SYSTEM MODEL

Data Collection: Data collection is the procedure of collecting, measuring, and analyzing accurate insights for research using standard validated techniques.

Input Data: We have taken a dataset as the input which contains various parameters that cause low birth weight for infants.

Data Cleaning: In this process, we have fixed or removed incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset. When combining multiple data sources, there are many opportunities for data to be duplicated or mislabeled. If data is incorrect, outcomes and algorithms are unreliable, even though they may look correct. So, data cleaning is an important step.

Splitting the Data: We have split the dataset into train and test sets to evaluate how well our machine learning model performs. The train set is used to fit the model, and the statistics of the train set are known. The second set is called the test data set, this set is solely used for predictions.

Training and Testing the Data: Training and Testing are done to measure the accuracy of our model.

The training dataset is used to fit the model, and the test dataset is used to evaluate the model.

Evaluation: It is done to know How well is my model doing. Will training my model on more data improve its performance? and Do I need to include more features?

Prediction: This involves certain manipulations of data from existing data sets with the goal of identifying some new trends and patterns. These trends and patterns are then used to predict future outcomes and trends regarding whether the baby will be born with a low weight or not.

4.2. 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 model 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 a 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.

UML is a very important part of developing object-oriented and software development processes. 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:

Provide users with a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.

Provide extensibility and specialization mechanisms to extend the core concepts.

Be independent of programming languages and development processes.

Provide a formal basis for understanding the modeling language.

Encourage the growth of OO in the tools market.

Support higher-level development concepts such as collaborations, frameworks, patterns, and components.

Integrate best practices.

USE CASE DIAGRAM

A Use Case Diagram in the Unified Modelling 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.

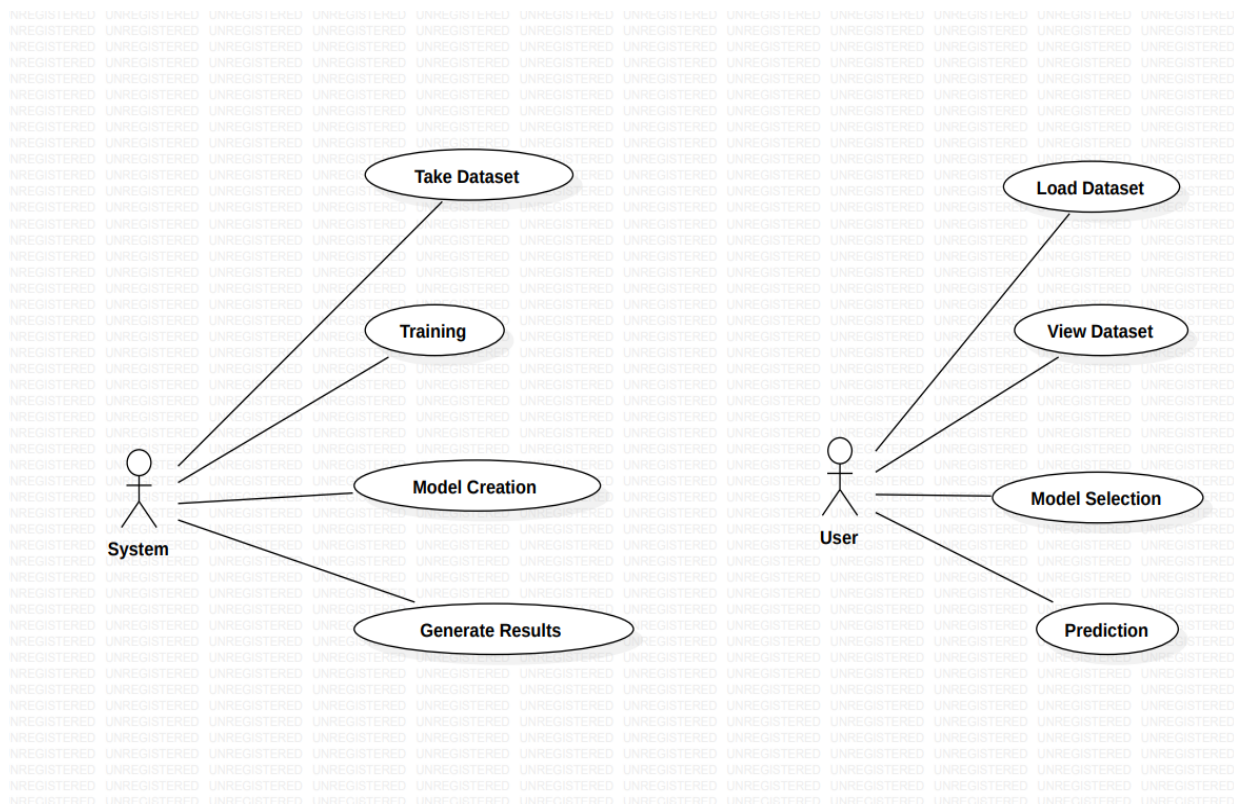
The main purpose of a use case diagram is to show what system functions are performed for which actor. The Roles of the actors in the system can be depicted. Interaction among actors is not shown on the use case diagram. If this interaction is essential to a coherent description of the desired behavior, perhaps the system or use case boundaries should be re-examined. Alternatively, interaction among actors can be part of the assumptions used in the use case.

Use cases:

A use case describes a sequence of actions that provide something of measurable value to an actor and is drawn as a horizontal ellipse.

Actors:

An actor is a person, organization, or external system that plays a role in one or more interactions with the system.



USECASE DIAGRAM

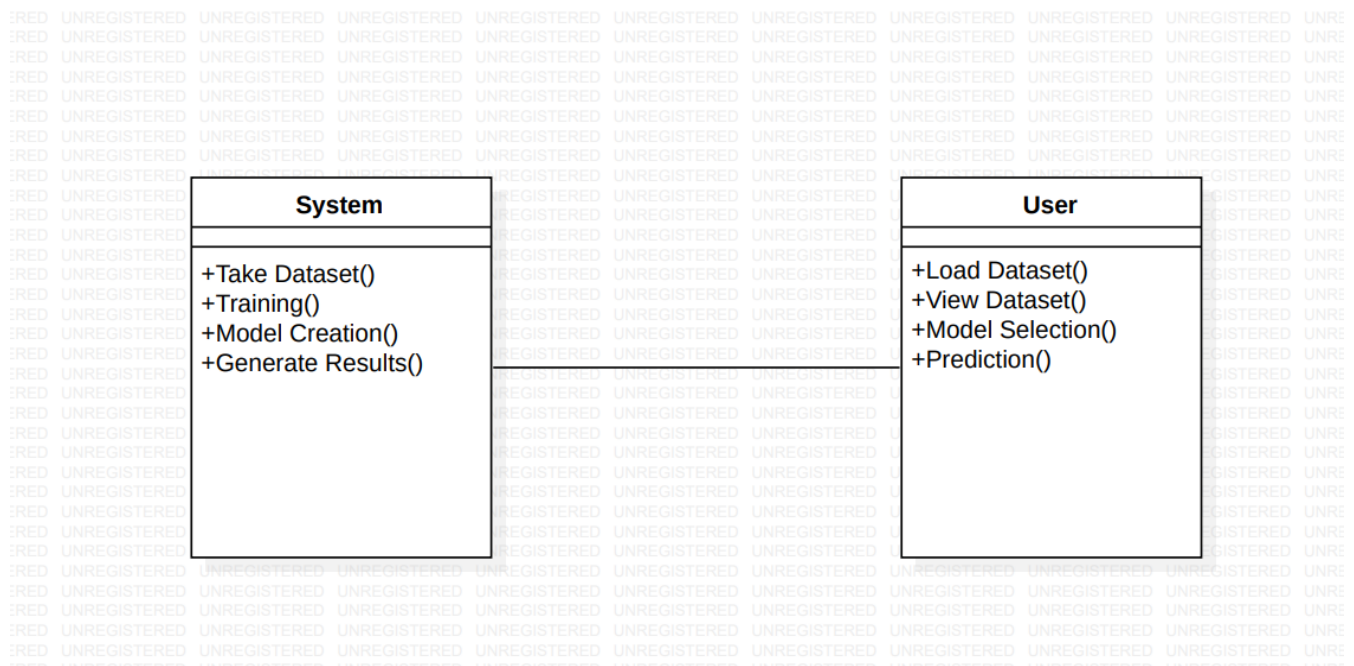
CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations, and relationships among objects. Purpose of Class Diagrams

- Shows the static structure of classifiers in a system
- Diagram provides a basic notation for other structure diagrams prescribed by UML
- Helpful for developers and other team members too
- Business Analysts can use class diagrams to model systems from a business

A UML class diagram is made up of:

- A set of classes
- A set of relationships between classes.



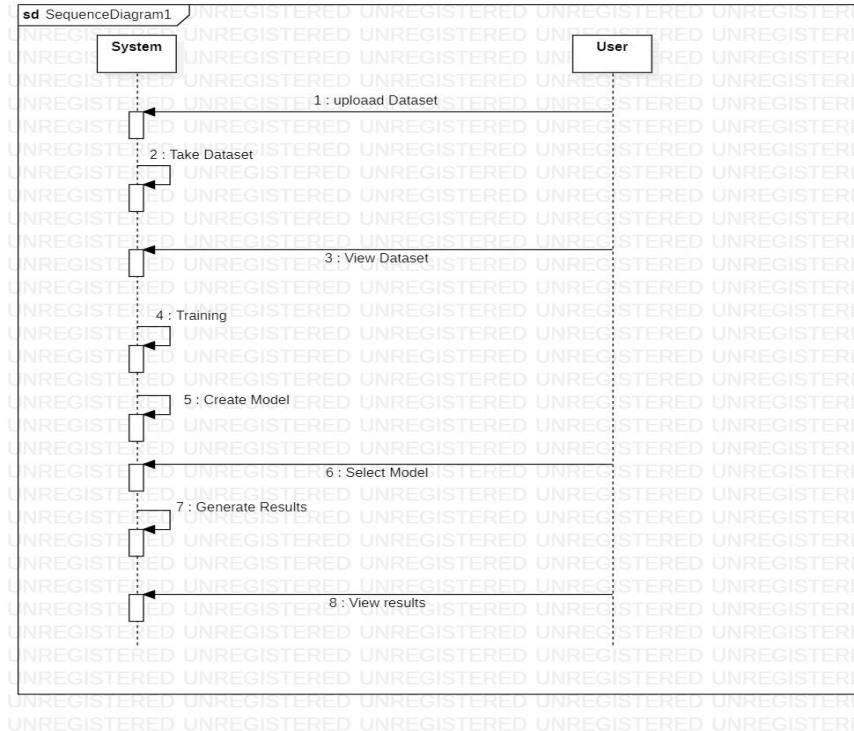
CLASS DIAGRAM

SEQUENCE DIAGRAM

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration.

Sequence Diagram captures:

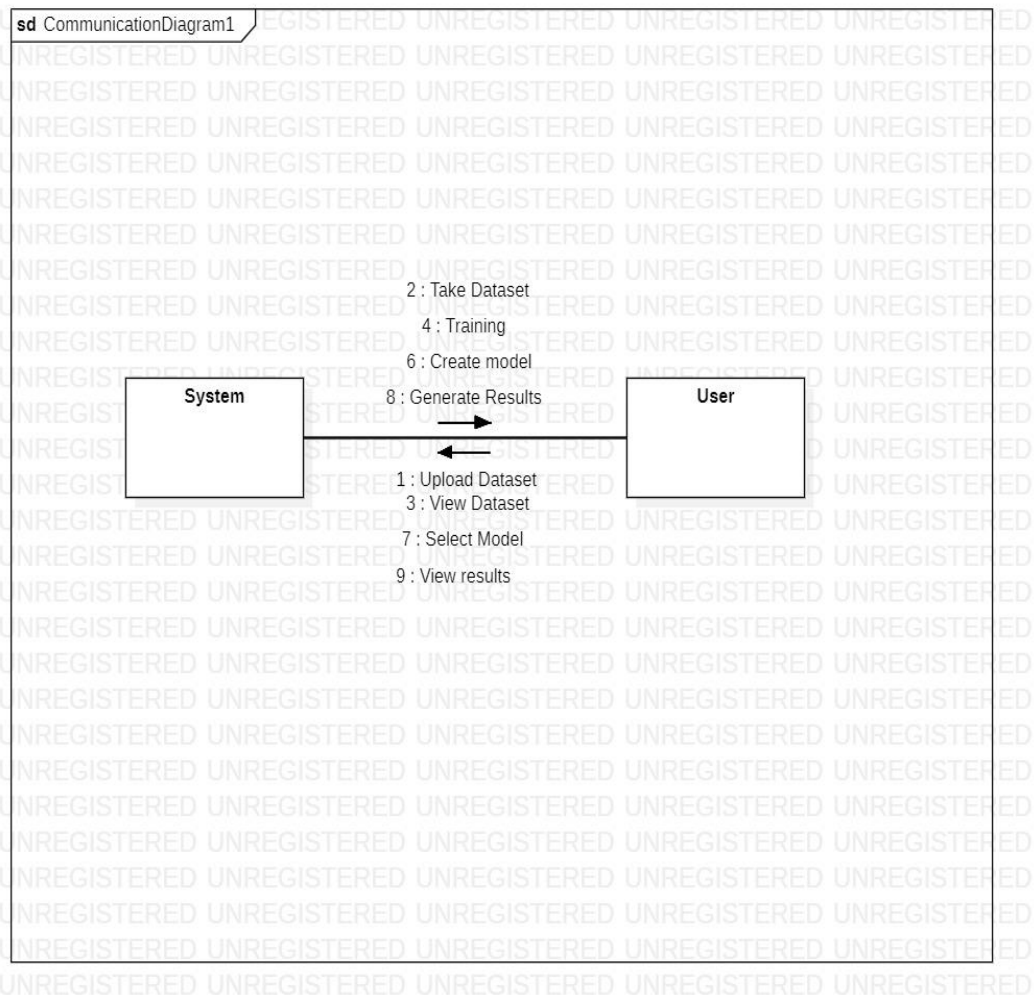
- The interaction that takes place in a collaboration that either realizes a use case or an operation.
- High-level interactions between the users of the system and the system, between the system and other systems, or between subsystems.
- Model the high-level interaction between active objects in a system.
- Model the interaction between object instances within a collaboration that realizes a use case.
- Model the interaction between objects within a collaboration that realizes an operation.
- Either model generic interactions or specific instances of interaction.



SEQUENCE DIAGRAM

COLLABORATION DIAGRAM

In the 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.



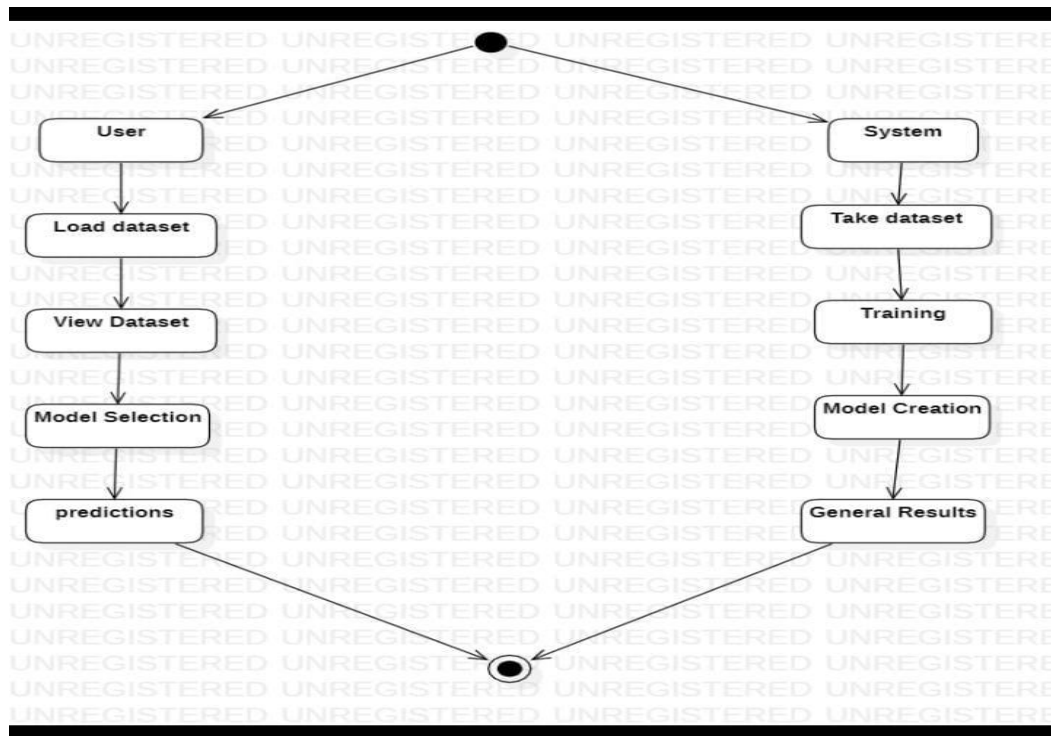
COLLABORATION DIAGRAM

ACTIVITY DIAGRAM

The activity diagram is another important behavioral diagram in the UML diagram to describe dynamic aspects of the system. An activity diagram is essentially an advanced version of a flow chart that models the flow from one activity to another activity.

Purpose of Activity Diagrams

- Draw the activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched, and concurrent flow of the system.



ACTIVITY DIAGRAM

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.



DEPLOYMENT DIAGRAM

COMPONENT DIAGRAM

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components 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.

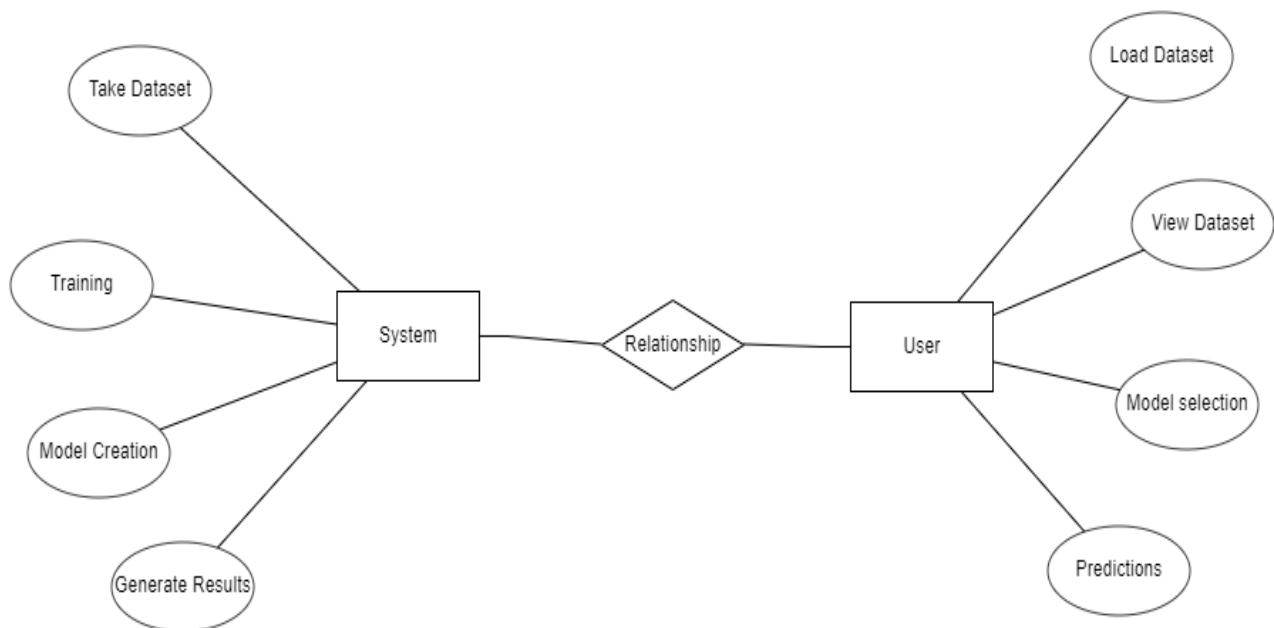


COMPONENT DIAGRAM

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.

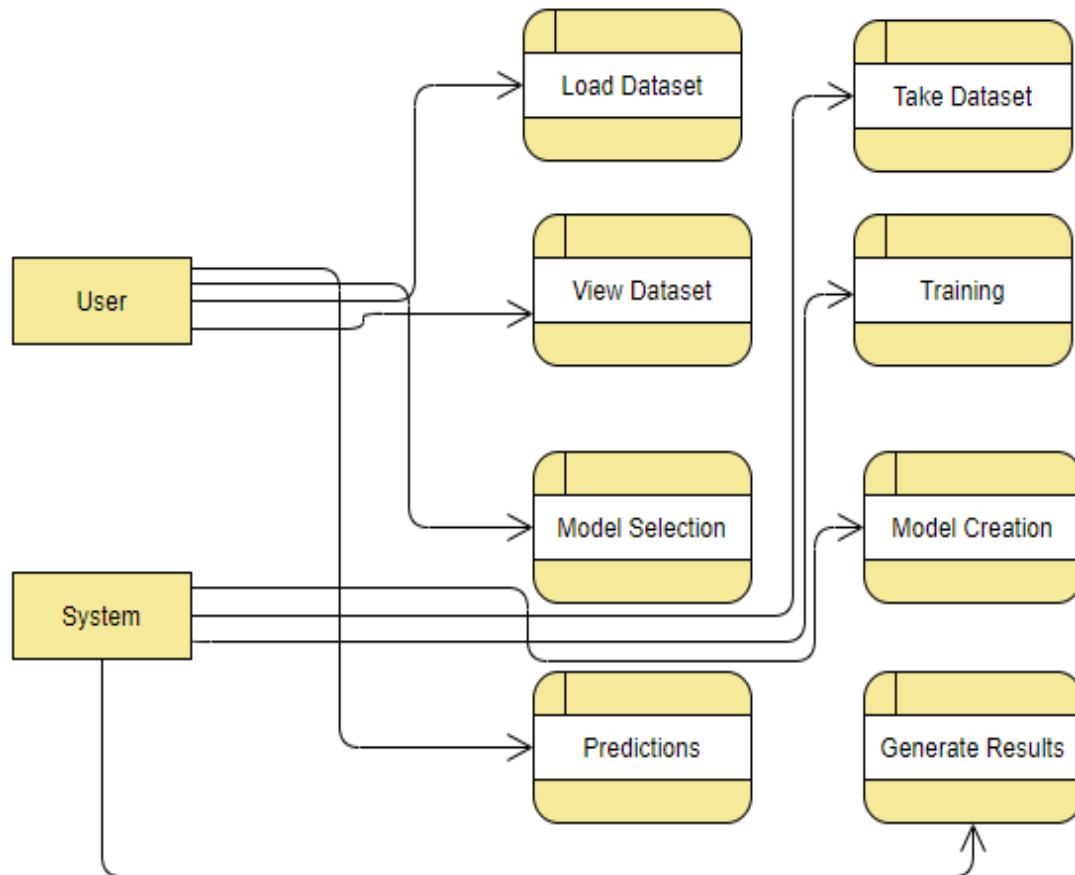
An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete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



ER DIAGRAM

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.



DFD DIAGRAM

METHODOLOGY

5. METHODOLOGY

MODULAR DIVISION

The entire architecture is divided into various modules:

1. System:

Takes Dataset:

The system allows users to upload the Dataset.

Store Dataset:

The System stores the dataset given by the user.

Model selection:

The system takes the data from the user and fed that data to the selected model.

Model Predictions:

The system takes the data given by the user and predicts the output based on the given data.

2. User:

Load Dataset:

The user can load the dataset he/she wants to work on.

View Dataset:

The User can view the dataset by clicking the view dataset module.

Select model:

User can select the model provided by the system for accuracy.

Predictions:

User can enter random values for prediction.

SYSTEM IMPLEMENTATION

6. SYSTEM IMPLEMENTATION

6.1. INTRODUCTION

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 a 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.

Step 1: Start with taking the input data.

Step 2: Cleaning and Preprocessing the data.

Step 3: Splitting the data into training and testing.

Step 4: Evaluating the train and test data.

Step 5: Predicting the data.

Step 6: Stop.

6.2. PROJECT MODULES:

Code snippet for importing libraries:

```
import os
import xgboost as xgb
import pandas as pd
from flask import Flask, render_template, request
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

Code snippet for connecting Flask:

```
app = Flask(__name__)
app.config['upload folder']='uploads'
@app.route('/')
def home():
    return render_template('index.html')
global path
```

Code snippet for uploading the data:

```
@app.route('/load data',methods=['POST','GET'])
def load_data():
    if request.method == 'POST':
        file = request.files['file']
        filetype = os.path.splitext(file.filename)[1]
        if filetype == '.csv':
            path = os.path.join(app.config['upload folder'], file.filename)
            file.save(path)
            print(path)
            return render_template('load data.html',msg = 'success')
        elif filetype != '.csv':
            return render_template('load data.html',msg = 'invalid')
        return render_template('load data.html')
    return render_template('load data.html')
```

Code snippet for viewing the data:

```
@app.route('/view data',methods = ['POST','GET'])
def view_data():
    file = os.listdir(app.config['upload folder'])
    path = os.path.join(app.config['upload folder'],file[0])
    global df
    df = pd.read_csv(path)
    print(df)
    return render_template('view data.html',col_name =df.columns.values,row_val =
list(df.values.tolist()))
```

Code Snippet for training and testing the data:

```
@app.route('/model',methods = ['POST','GET'])
def model():
    if request.method == 'POST':
        global scores1,scores2,scores3,scores4
        global df
        filename = os.listdir(app.config['upload folder'])
        path = os.path.join(app.config['upload folder'],filename[0])
        df = pd.read_csv(path)
        global testsize
        testsize =int(request.form['testing'])
        print(testsize)
        global x_train,x_test,y_train,y_test
        testsize = testsize/100
        print(df)
        X = df.drop(['result'],axis = 1)
        y = df.result
        x_train,x_test,y_train,y_test = train_test_split(X,y,test_size=testsize,random_state=0)
```

Code Snippet for finding the Accuracy:

```
model = int(request.form['selected'])
    if model == 1:
        dtc = DecisionTreeClassifier()
        model1 = dtc.fit(x_train,y_train)
        pred1 = model1.predict(x_test)
        scores1 = accuracy_score(y_test,pred1)
        return render_template('model.html',score = round(scores1,4),msg =
'accuracy',selected = 'DECISION TREE CLASSIFIER')
    elif model == 2:
        rfc = RandomForestClassifier()
        model2 = rfc.fit(x_train,y_train)
        pred2 = model2.predict(x_test)
        scores2 =accuracy_score(y_test,pred2)
```

```
return render_template('model.html',msg = 'accuracy',score = round(scores2,3),selected

= 'RANDOM FOREST CLASSIFIER')
elif model == 3:
    svc = SVC()
    model3 = svc.fit(x_train,y_train)
    pred3 = model3.predict(x_test)
    scores3 = accuracy_score(y_test,pred3)
    return render_template('model.html',msg = 'accuracy',score = round(scores3,3),selected
    = 'SUPPORT VECTOR CLASSIFIER ')
elif model == 4:
    xgbc = xgb.XGBClassifier()
    model4 = xgbc.fit(x_train,y_train)
    pred4 = model4.predict(x_test)
    scores4 = accuracy_score(y_test,pred4)
    return render_template('model.html',msg = 'accuracy',score = round(scores4,3), selected
    = 'XGBOOST CLASSIFIER')
return render_template('model.html')
```

Code snippet for finding the best algorithm:

```
import pandas as pd
import seaborn
import sns as sns
from sklearn import svm
from sklearn.tree import DecisionTreeClassifier
import matplotlib.pyplot as plt!
df = pd.read_csv('Z:\\2022\\GROWTH PROJECTS\\---Low Birth Weight Predictor---
\\cleaned.csv')
df.head()
df.isnull().sum()
df.info()
X = df.drop(['result'],axis = 1)
y = df.result
from sklearn.metrics import accuracy_score, precision_score, recall_score,
classification_report
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size =0.2,random_state = 20)
```

```
from sklearn.ensemble import RandomForestClassifier  
dtc = DecisionTreeClassifier()
```

```
model2 = dtc.fit(x_train,y_train)  
pred2 = model2.predict(x_test)  
a = accuracy_score(y_test,pred2)  
rfc = RandomForestClassifier()  
model = rfc.fit(x_train,y_train)  
pred = model.predict(x_test)  
b = accuracy_score(y_test,pred)  
import xgboost as xgb  
xgbc = xgb.XGBClassifier()
```

```
model1 = xgbc.fit(x_train,y_train)  
# print(model1.feature_importances_)  
pred1 = model1.predict(x_test)  
d = accuracy_score(y_test,pred1)
```

```
svc = svm.SVC()  
model3 =svc.fit(x_train,y_train)  
pred3 = model3.predict(x_test)  
c = accuracy_score(y_test,pred3)
```

```
data = pd.DataFrame({'Models': ['Decision Tree', 'Random Forest', 'Support vector  
machine', "XGBoost"],  
'Accuracy': [a * 100,b * 100, c * 100, d* 100]})  
print(data)  
ch = seaborn.barplot(x = data.Models,y= data.Accuracy)  
# f = seaborn.distplot(data.Accuracy)  
plt.show()
```

Code snippet for Predicting:

```
@app.route('/prediction',methods = ['POST',"GET"])  
def prediction():
```



```
if request.method == 'POST':

    a = float(request.form['a'])
    b = float(request.form['b'])
    c = float(request.form['c'])
    d = float(request.form['d'])
    e = float(request.form['e'])
    f = float(request.form['f'])
    g = float(request.form['g'])
    h = float(request.form['h'])
    i = int(request.form['i'])

    values = [[float(a),float(b),float(c),float(d),float(e),float(f),float(g),float(h),float(i)]]
    dtc = DecisionTreeClassifier()
    model = dtc.fit(x_train,y_train)
    pred = model.predict(values)
    return render_template('prediction.html',msg='success',result = pred)
    return render_template('prediction.html')

if __name__ == '__main__':
    app.run(debug=True)
```

6.3. SCREENS:

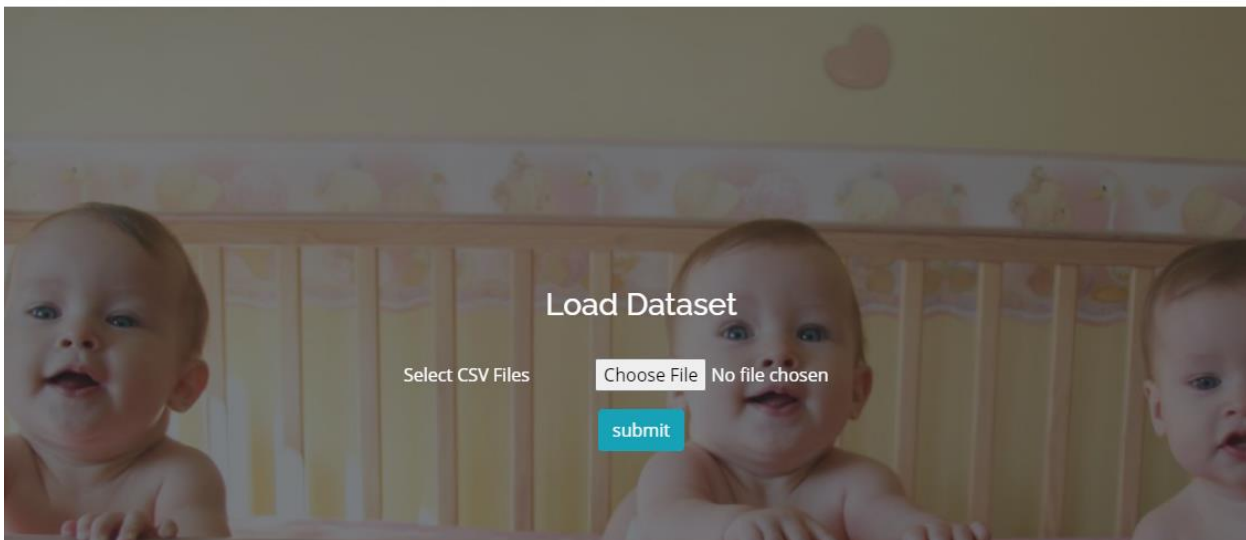
User Interface

LOW BIRTH WEIGHT GROUP PREDICTION [Home](#) [Load Data](#) [View Data](#) [Select Model](#) [Prediction](#)



Screen: 1 – Home Page

LOW BIRTH WEIGHT GROUP PREDICTION [Home](#) [Load Data](#) [View Data](#) [Select Model](#) [Prediction](#)



Screen: 2 – Data Loading Page

LOW BIRTH WEIGHT GROUP PREDICTION [Home](#) [Load Data](#) [View Data](#) [Select Model](#) [Prediction](#)

S/N	community	age	weight1	history	HB	IFA	BP1	education	res	result
1	1.0	26.0	37.0	1.0	5.9	1.0	1.4444444440000002	5.0	1.0	0.0
2	1.0	21.0	42.0	1.0	9.2	1.0	1.375	5.0	1.0	0.0
3	1.0	21.0	47.136364	1.0	8.8	1.0	1.5	5.0	1.0	0.0
4	1.0	21.0	47.136364	1.0	9.2	1.0	2.125	5.0	1.0	0.0
5	1.0	21.0	47.136364	1.0	8.0	1.0	1.375	5.0	1.0	0.0
6	1.0	24.0	33.0	1.0	9.3	1.0	1.571	5.0	1.0	0.0
7	1.0	26.0	35.0	1.0	9.2	1.0	1.571428571	5.0	1.0	0.0
8	4.0	26.0	31.0	1.0	9.076922999999999	1.0	1.625	5.0	1.0	0.0
9	3.0	21.0	47.136364	1.0	11.0	1.0	1.375	5.0	1.0	0.0
10	1.0	22.0	30.0	1.0	9.0	1.0	1.482	5.0	1.0	0.0
11	4.0	17.0	30.0	1.0	9.0	0.0	1.375	5.0	1.0	0.0
12	3.0	35.0	54.0	1.0	9.9	1.0	1.571428571	5.0	1.0	0.0

Screen: 3 – Data Viewing Page

LOW BIRTH WEIGHT GROUP PREDICTION [Home](#) [Load Data](#) [View Data](#) [Select Model](#) [Prediction](#)

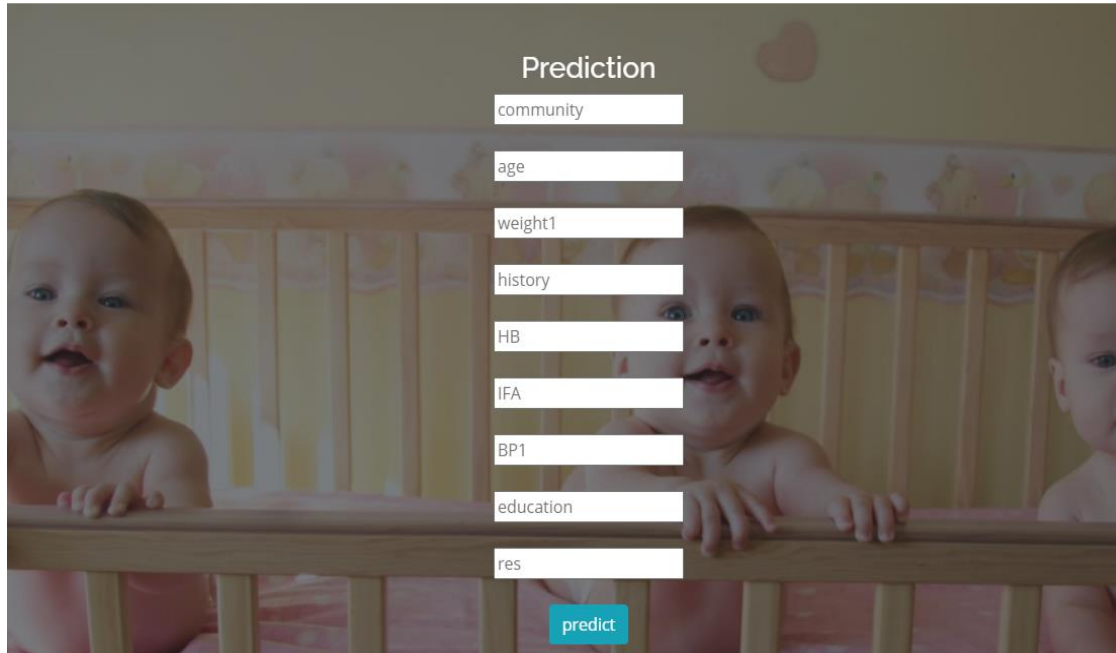
Model Selection

Select Testsize

Select Model

Screen: 4 – Model Selection Page

LOW BIRTH WEIGHT GROUP PREDICTION

[Home](#)[Load Data](#)[View Data](#)[Select Model](#)[Prediction](#)

Prediction

community

age

weight1

history

HB

IFA

BP1

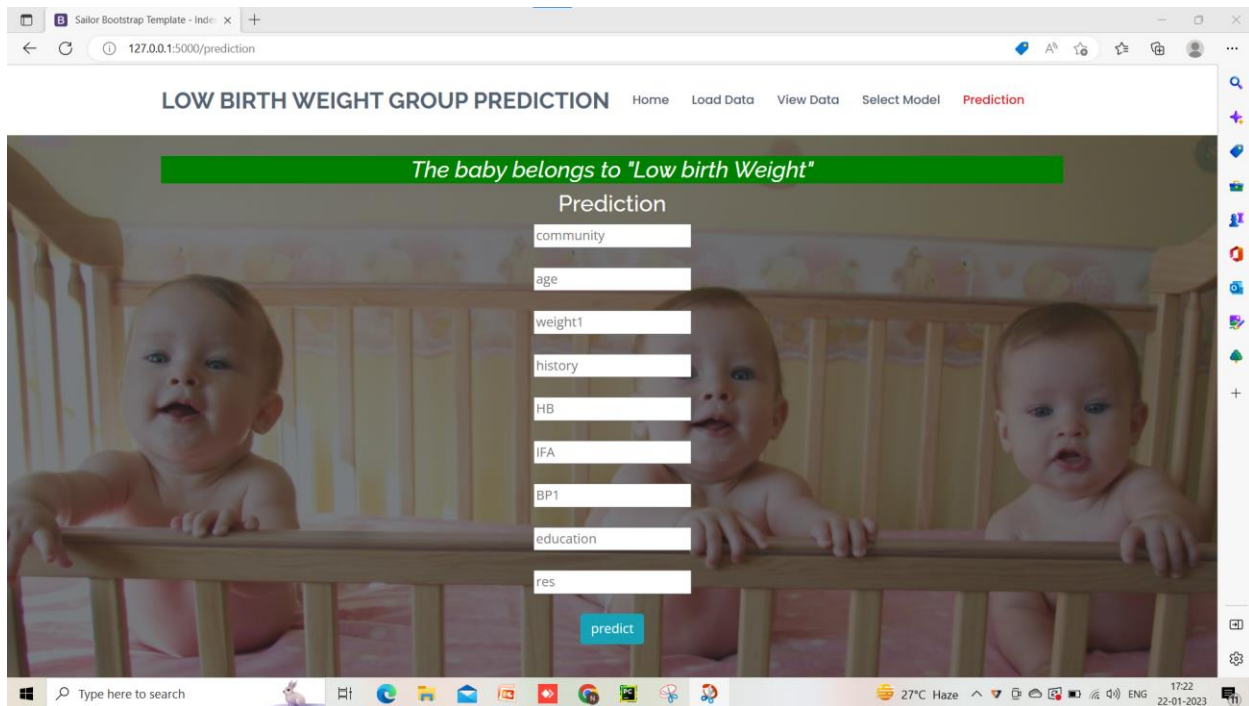
education

res

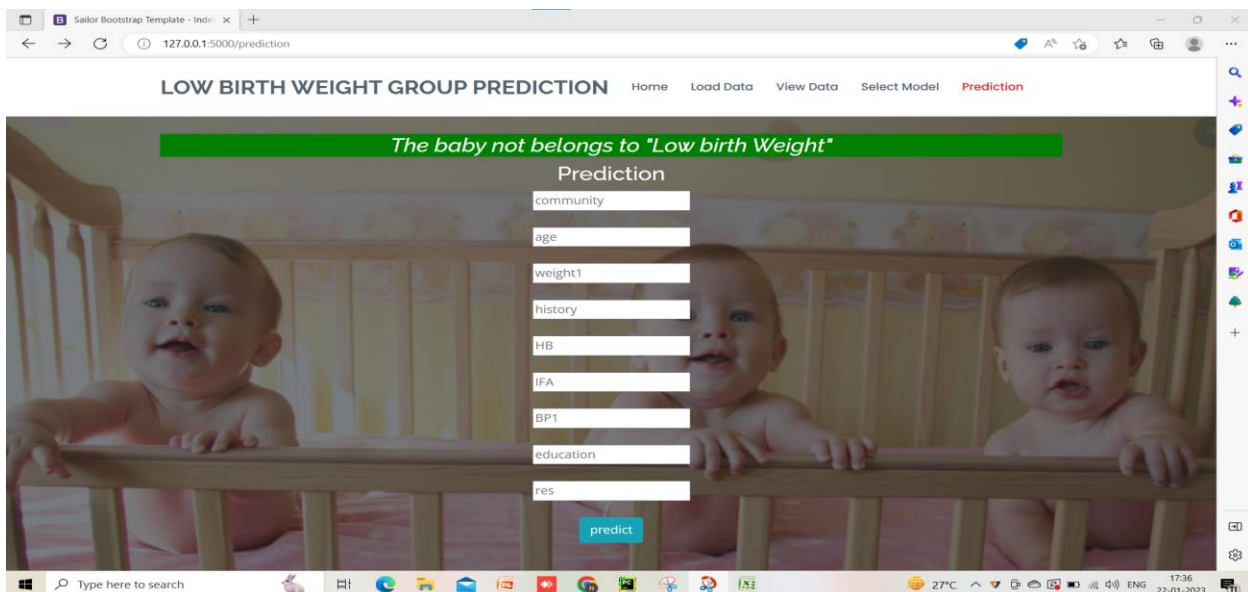
predict

Screen: 5 – Prediction Page

FINAL OUTPUT:



SCREEN: 6- LOW-BIRTH-WEIGHT CASE



SCREEN: 7- NOT LOW-BIRTH-WEIGHT CASE

SYSTEM TESTING

7.SYSTEM TESTING

7.1. INTRODUCTION:

Testing is the process of finding differences between the expected behavior specified by the system models and the observed behavior of the system.

Software Testing is an important element of the software quality assurance and represents the ultimate review of specification, design and coding. The increasing feasibility of software as a system and the cost associated with the software failures are motivated forces for III planned through testing.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The process of software testing aims not only at finding faults in the existing software but also at finding measures to improve the software in terms of efficiency, accuracy and usability.

Software Testing can be broadly classified into two types. They are:

1. **Manual Testing:** Manual testing includes testing software manually, i.e., without using any automated tool or any script. In this type, the tester takes over the role of an end-user and tests the software to identify any unexpected behaviour or bug.
2. **Automation Testing:** Automation testing, which is also known as Test Automation, is when the tester writes scripts and uses other software to test the product. This process involves automation of a manual process. Automation Testing is used to re-run the test scenarios that were performed manually, quickly, and repeatedly.

TESTING OBJECTIVES

These are several rules that can save as testing objectives:

- Testing is a process of executing a program with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- To ensure that during operation the system will perform as per specification.
- To make sure that the system meets the user requirements during operation.

- To make sure that during the operation, incorrect input, processing, and output will be detected.
- To see that when correct inputs are fed to the system, the outputs are correct.
- To verify that the controls are incorporated in the same system as intended.

TEST LEVELS

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or darkness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies, and/or a finished product. Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

TESTING ACTIVITIES

- Inspecting a component, finds faults in an individual component through the manual inspection of its source code.
- Unit testing, which finds faults by isolating an individual component using test stubs and drivers and by exercising the components using test cases.
- Integration testing, which finds faults by integrating several components together.
- System testing, which focuses on the complete system and its functional requirements and non-functional requirements and its target environment.

Testing is the phase where the errors remaining from all the previous phases must be detected. Hence testing performs a very critical role in quality assurance and in ensuring the reliability of software by providing the software with a set of test outputs and observing if the software behaves as expected, then the conditions under which the failure occurs or is needed for debugging and correction.

Error

The term error is used in two different ways. It refers to the disciplinary between a computed, observed, or measured value and a true, specified, or theoretically correct value. Error is also used to refer to human actions that result in software containing a fault. This definition is quite general and encompasses all the phases.

Failure

Failure is the inability of the system or the components to perform a required function according to its specifications. In other words, failure is the manifestation of error. But the mere presence of an error may not cause a failure presence of an error implies that a fault must be present in the system. However, the presence of a default does not imply that a failure must occur.

A test case is a triplet $[i, s, o]$ where i stand for the data input to the system, s is the state of the system at which the data is input, and o is the expected output of the system. A test suite is the set of all test cases with which a given software product is to be tested.

Verification and Validation

Verification is the process of determining whether one phase of the software product confirms its previous. Verification is concerned with the phase containing the errors.

7.2. TESTING METHODS

A testing strategy is a roadway, giving how to conduct a test. Our testing strategy is flexible enough to promote customization that is necessary for the due course of the development process. For instance, during code, we find a change in design. We maintain a change log and refer to it at the appropriate time during testing. The first approach is what is known as black box testing and the second is called white box testing. We'll be using a mixed approach, more popularly known as sandwich testing. We apply white box testing techniques to ascertain the functionalities top-down and then we use black box testing to demonstrate everything that runs as expected.

A strategy for testing integrates software test case design methods into a well-planned series of steps that result in the construction of software. The objective of the testing is to reduce risks inherent in computer software.

The various testing methods are:

- White box testing
- Black box testing

White Box Testing

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

White box testing is a testing case design method that uses the control structure of the procedure design to derive test cases. All independent paths in a module are exercised at least once, all logical decisions are exercised at once, execute all loops at boundaries and within their operational bounds exercise internal data structure to ensure their validity. Here, the customer is given three chances to enter a valid choice from the given menu. After which the control exits the current menu.

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, like 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.

Black box testing attempts to find errors in the following areas or categories, incorrect or missing functions, interface errors, errors in data structures, performance errors, and initialization and termination errors. Here all the input data must match the data type to become a valid entry.

7.3. TESTING TYPES

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.

Unit testing is essential for the verification of the code produced during the coding phase and the goal is to test the internal logic of the module or program. In the generic code project, the unit testing is done during the coding phase of data entry forms whether the functions are

working properly or not. In this phase, all the drivers are tested whether they are rightly connected or not.

Integration Testing

Integration tests are designed to test integrated software components to determine if they run as one program. Testing is event-driven and is more concerned with the basic outcome of screens or fields.

All the tested modules are combined into sub-systems, which are then tested. The goal is to see if the modules are properly integrated, and the emphasis is on the testing interfaces between the modules. In the generic code integration testing is done mainly on the table creation module and insertion module.

Validation Testing:

This testing concentrates on confirming that the software is error-free in all aspects. All the specified validations are verified and the software is subjected to hardcore testing. It also aims to determine the degree of deviation that exists in the software design from the specification; they are listed out and corrected.

System Testing:

System tests are designed to validate a fully developed system with a view to assure that it meets specified requirements.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user.

7.4. TEST CASES

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.

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.

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 were encountered.

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 were encountered.

CONCLUSION

8. CONCLUSION

In this application, we have successfully created a Machine Learning model to estimate whether the baby belongs to the Low Birth Weight or not belongs to 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, and Support Vector Classifier Decision Tree Classifier performs well with better accuracy.

BIBLIOGRAPHY

9. BIBILIOGRAPHY

References:

- [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 Organization 2004, Low Birth Weight: Country, regional and global estimates, New York, UNICEF.

Textbooks:

[1] Machine Learning: Li Kanghua, Jiang Shan. Machine Learning and Cultural Production Reform Based on the Perspective of the Development of AI Technology [J]. Journal of Xiangtan University (Philosophy and Social Sciences), 2020, 44 (01): 74-79.

[2] Python: Anuag. (2020). Future of Python in the industry. Available at: <https://www.newgenaps.com/blog/future-of-python-in-the-industry/>. Accessed on 20 June, 2020.

Websites:

LBW Study: https://en.wikipedia.org/wiki/Low_birth_weight

UML Diagrams: <https://creately.com/lp/uml-diagram-tool/>

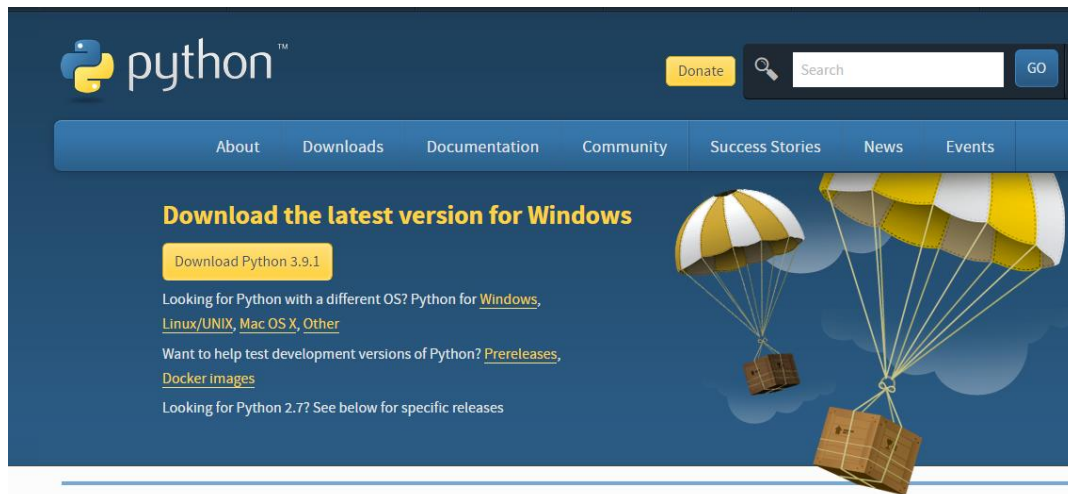
APPENDIX

10. APPENDIX

10.1. SOFTWARE INSTALLATION FOR THIS PROJECT

Installing Python:

1. To download and install Python visit the official website of Python <https://www.python.org/downloads/> and choose your version.



PYTHON INSTALLATION

2. Once the download is complete, run the exe for installing Python. Now click on Install Now.
3. You can see Python installed at this point.
4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

Installing PyCharm:

1. Visit the website <https://www.jetbrains.com/pycharm/download/> and click the "DOWNLOAD" link under the Community Section.

Download PyCharm

[Windows](#)

[Mac](#)

[Linux](#)

Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

Download

Free trial

Community

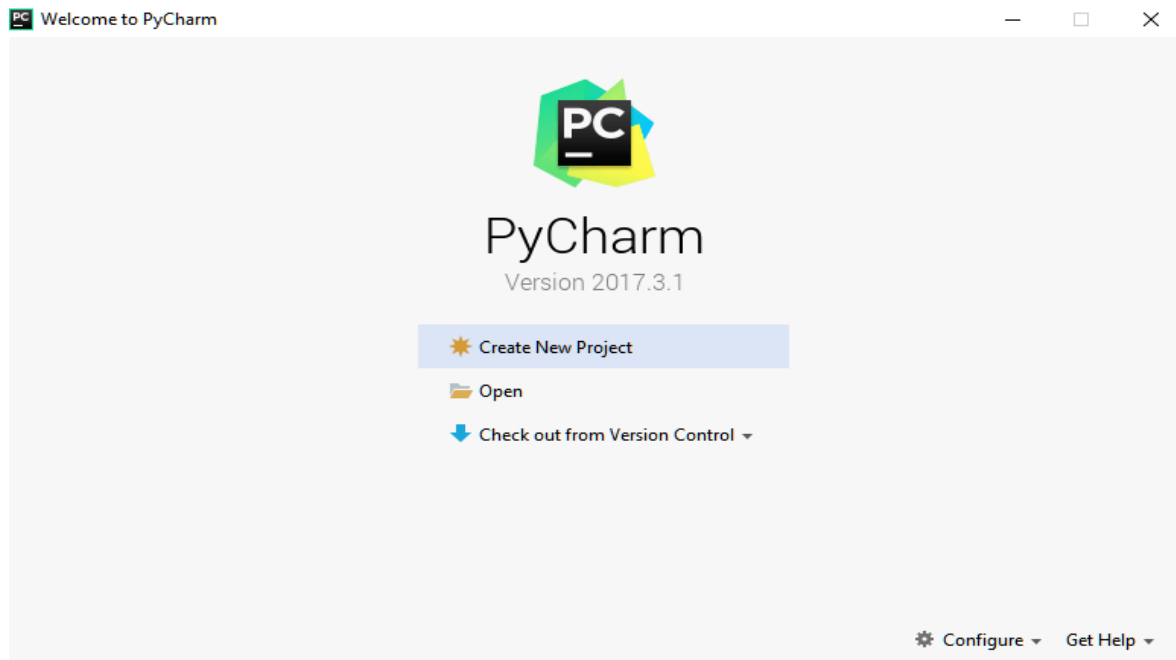
For pure Python development

Download

Free, open-source

DOWNLOADING PyCharm

1. Once the download is complete, run the exe for installing PyCharm. The setup wizard should have started. Click "Next".
2. On the next screen, Change the installation path if required. Click "Next".
3. On the next screen, you can create a desktop shortcut if you want and click on "Next".
4. Choose the start menu folder. Keep selecting JetBrains and click on "Install".
5. Wait for the installation to finish.
6. Once installation is finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the "Run PyCharm Community Edition" box first and click "Finish".
7. After you click on "Finish," the Following screen will appear.
8. You need to install some packages to execute your project in a proper way.
9. Open the command prompt/ anaconda prompt or terminal as administrator.
10. The prompt will get open, with a specified path, type "pip install package name" which you want to install (like NumPy, pandas, seaborn, scikit-learn, matplotlib, pyplot)
Ex: pip install numpy



CREATING PROJECT IN PyCharm

```
C:\WINDOWS\system32>pip install numpy==1.18.5
Collecting numpy==1.18.5
  Downloading numpy-1.18.5-cp36-cp36m-win_amd64.whl (12.7 MB)
    | 12.7 MB 939 kB/s
ERROR: tensorboard 2.0.2 has requirement setuptools>=41.0.0, but
Installing collected packages: numpy
Successfully installed numpy-1.18.5
```

INSTALLING PACKAGES

10.2. INTRODUCTION TO PYTHON

Python is an open-source, high-level programming language developed by Guido van Rossum in the late 1980s and presently administered by Python Software Foundation. It came from the ABC language that he helped create early on in his career. Python is a powerful language that you can use to create games, write GUIs, and develop web applications.

It is a high-level language. Reading and writing codes in Python is much like reading and writing regular English statements. Because they are not written in the machine-readable language, Python programs need to be processed before machines can run them. Python is an interpreted language.

Readability: Python programs use clear, simple, and concise instructions that are easy to read even by those who have no substantial programming background. Programs written in Python are, therefore, easier to maintain, debug, or enhance.

Higher productivity: Codes used in Python are considerably shorter, simpler, and less verbose than other high-level programming languages such as Java and C++. In addition, it has well-designed built-in features and a standard library as well as access to third party modules and source libraries.

Less learning time: Python is relatively easy to learn. Many find Python a good first language for learning programming because it uses simple syntax and shorter codes. Python works on Windows, Linux/UNIX, Mac OS X, other operating systems, and small form devices.

10.3. SOME PYTHON LIBRARIES

Some Python Libraries:

1. Pandas
2. Numpy
3. Pymysql
4. Random

Pandas:

- Pandas provide us with many Series and Data Frames. It allows you to easily organize, explore, represent, and manipulate data.
- Smart alignment and indexing featured in Pandas offer you perfect organization and data labeling.
- Pandas have some special features that allow you to handle missing data or values with proper measures.
- This package offers you such a clean code that even people with no or basic knowledge of programming can easily work with it.
- It provides a collection of built-in tools that allows you to both read and write data in different web services, data-structure, and databases as well.
- Pandas can support JSON, Excel, CSV, HDF5, and many other formats. In fact, you can merge different databases at a time with Pandas.

Numpy:

- Arrays of Numpy offer modern mathematical implementations on huge amount of data. Numpy makes the execution of these projects much easier and hassle-free.
- Numpy provides masked arrays along with general array objects. It also comes with functionalities such as manipulation of logical shapes, discrete Fourier transform, general linear algebra, and many more.
- While you change the shape of any N-dimensional arrays, Numpy will create new arrays for that and delete the old ones.
- This python package provides useful tools for integration. You can easily integrate Numpy with programming languages such as C, C++, and FORTRAN code.
- Numpy provides such functionalities that are comparable to MATLAB. They both allow users to get faster with operations.

Pymysql:

- PyMySQL is a database connector for Python, libraries to enable Python programs to talk to a MySQL server.
- Access to the port settings through Python properties.
- PyMySQL is a pure Python MySQL driver, first written as a rough port of the MySQL-Python driver.
- PyMySQL meets all the criteria for a driver.
- It is fully open source, hosted on Github, released on Pypi, and actively maintained.
- It is written in pure Python so is event let-monkey patch is compatible and is fully Python 3 compatible.

Random:

- The random module is a built-in module to generate the pseudo-random variables.
- It can be used to perform some actions randomly such as getting a random number, selecting random elements from a list, shuffling elements randomly, etc.
- Generate random numbers for various distributions including integers and floats.
- Random Sampling and choosing elements from the population.
- Functions of the random module.
- Shuffle the sequence data. Seed the random generator.
- Generate random strings and passwords

Python Namespace:

- A **namespace** (sometimes also called a context) is a naming system for making names unique to avoid ambiguity. Everybody knows a name spacing system from daily life, i.e. the naming of people by a first name and family name (surname). An example is a network: each network device (workstation, server, printer,) needs a unique name and address. Yet another example is the directory structure of file systems.
- The same file name can be used in different directories, the files can be uniquely accessed via the pathnames. Many programming languages use namespaces or contexts for identifiers. An identifier defined in a namespace is associated with that namespace.

10.4. WHY PYTHON FOR MACHINE LEARNING?

1. Python is easy to understand.

Python is the most suitable programming language for this because it is easy to understand, and you can read it for yourself. Its readability, non-complexity, and ability for fast prototyping make it a popular language among developers and programmers around the world.

2. Python comes with many libraries.

Some of the libraries are:

- **Sci-kit-learn** for data mining, analysis, and Machine Learning.
- **TensorFlow**, a high-level neural network library.
- **pylearn2** which is also ideal for data mining and Machine Learning, but more flexible than sci-kit-learn.

3. Python allows easy and powerful implementation.

With other programming languages, coding beginners or students need to familiarize themselves with the language first before being able to use it for ML or AI.

This is not the case with Python. Even if you only have basic knowledge of the Python language, you can already use it for Machine Learning because of the huge amount of libraries, resources, and tools available to you.

4. Friendly syntax and human-level readability

Python is an object-oriented programming language that uses modern scripting and friendly syntax.

5. Community

Lastly, Python provides broad support. Because a lot of people, both programmers and average users, view Python as a standard, its support community is huge, increasing Python's popularity even more.

10.5. INTRODUCTION TO FLASK FRAMEWORK

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require tools or libraries.[2] It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies, and several common framework-related tools.

Unlike the Django framework, Flask is very Pythonic. It's easy to get started with Flask because it doesn't have a huge learning curve. On top of that, it's very explicit, which increases readability. To create the "Hello World" app, you only need a few lines of code.

This is a boilerplate code example.

```
from flask import Flask

app = Flask(__name__)

@app.route('/')

def hello_world():

    return 'Hello World!'

if __name__ == '__main__':

    app.run ()
```

If you want to develop on your local computer, you can do so easily. Save this program as server.py and run it with python server.py.

```
$ python server.py

* Serving Flask app "hello"

* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

It then starts a web server which is available only on your computer. In a web browser open localhost on port 5000 (the URL) and you'll see "Hello World" show up.

To host and develop online, you can use PythonAnywhere

10.6. MACHINE LEARNING ALGORITHMS:

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.

Random Forest:

First, the Random Forest algorithm is a supervised classification algorithm. We can see it from its name, which is to create a forest in 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 the information gain or gain index approach.

The author gives four advantages to illustrate why we use the 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 the 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.

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 represent decisions and decision visually and explicitly 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, 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 a learning decision tree from data and the above tree is called a Classification tree as the target is to classify passengers as survived or died. Regression trees are represented in the same manner, just they predict continuous values like the price of a house. In general, Decision Tree algorithms are referred to as CART or Classification and Regression Trees.

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 set 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 run 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.