A Mini Project report

on

” CHILD MORTALITY PREDICTION USING MACHINE LEARNING ALGORITHM”

Submitted in Partial Fulfillment of the Academic

Requirement for the Award of Degree of

BACHELOR OF TECHNOLOGY

in

Computer Science and Engineering

Submitted by

B. Pruthvi	**(20R01A05K4)**
T. Dhanush Reddy	**(20R01A05P3)**
V. Sai Shyam Reddy	**(20R01A05P5)**

Under the esteemed guidance of

Mrs. B. Sunitha Devi

(Assistant Professor, Computer Science and Engineering Dept)

CMR INSTITUTE OF TECHNOLOGY

(UGC AUTONOMUS)

(Approved by AICTE,Affiliated to JNTU,Kukatpally,Hyderabad)

Kandlakoya,Medchal Road,Hyderabad

2022-2023

CMR GROUP OF INSTITUTIONS
EXPLORE TO INVENT

CMR INSTITUTE OF TECHNOLOGY

(UGC AUTONOMUS)

(Approved by AICTE,Affiliated to JNTU,Kukatpally,Hyderabad)

Kandlakoya,Medchal Road,Hyderabad

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that a Mini Project entitled with "CHILD MORTALITY PREDICTION USING MACHINE LEARNING TECHNIQUES" is being

Submitted by:

B. Pruthvi	**(20R01A05K4)**
T. Dhanush Reddy	**(20R01A05P3)**
V. Sai Shyam Reddy	**(20R01A05P5)**

In partial fulfilment of the requirement for award of the degree of B. Tech in CSE to the JNTUH, Hyderabad is a record of a Bonafide work carried out under our guidance and supervision.
The results in this project have been verified and are found to be satisfactory. The results embodied in this work have not been submitted to have any other University for award of any other degree or diploma.

Signature of Guide	**Signature of External**	**Signature of HOD**
Mrs. B. Sunitha Devi		**Mr. A. Prakash**
(Assistant Professor)		(Head of Department)

ACKNOWLEDGEMENT

We are extremely grateful to **Dr. M. Janga Reddy**, **Director**, **Dr. B. Satyanarayana**, **Principal** and M**r.A.Prakash**, **Head of Department**, Dept of Computer Science and Engineering, CMR Institute of Technology for their inspiration and valuable guidance during entire duration.

We are extremely thankful to **Mrs. B. Sunitha Devi, Assistant professor**, Dept of Computer Science and Engineering, CMR Institute of Technology for their constant guidance, encouragement and moral support throughout the project.

We express our thanks to all staff members and friends for all the help and coordination extended in bringing out this Project successfully in time.

Finally, we are very much thankful to our parents and relatives who guided directly or indirectly for successful completion of the project.

B. Pruthvi	**(20R01A05K4)**
T. Dhanush Reddy	**(20R01A05P3)**
V. Sai Shyam Reddy	**(20R01A05P5)**

ABSTRACT

Children's Mortality alludes to mortality of children younger than 5. The kid death rate, in addition under-five death rate, alludes to the probability of biting the mud among birth and exactly 5 years recent. The mortality of kids in addition happens in embryo. The purpose is to analysis AI based mostly strategies for grouping of mortality vertebrate upbeat characterization brings concerning best truth. The examination of dataset by directed AI procedure (SMLT) to catch a couple of data's like, variable characteristic proof, uni-variate investigation, bi-variate and multi-variate examination, missing value medicines and dissect the data approval, data cleaning/getting prepared and knowledge illustration are done on the entire given dataset. Our examination provides a whole manual for responsiveness investigation of model boundaries on execution within the characterization of vertebrate upbeat. To propose AN AI based mostly and moreover, to seem at and examine the presentation of various AI calculations for the given dataset..

INDEX

LIST OF FIGURES

LIST OF SCREENSHOTS

1. INTRODUCTION

A. Data Science

Information science is an interdisciplinary field that utilizes logical strategies, cycles, calculations and frameworks to separate information and experiences from organized and unstructured information, and apply information and noteworthy bits of knowledge from information across an expansive scope of use spaces. The expression "information science" has been followed back to 1974, when Peter Naur proposed it as an elective name for software engineering. In 1996, the International Federation of Classification Societies turned into the primary gathering to highlight information science as a subject explicitly. In any case, the definition was still in transition. The expression "information science" was first authored in 2008 by D.J. Patil, and Jeff Hammerbacher, the trailblazer leads of information and investigation endeavors at LinkedIn and Facebook. In under 10 years, it has become one of the most sultry and most moving callings on the lookout. Information science is the field of study that joins area aptitude, programming abilities, and information on math and measurements to separate significant bits of knowledge from information. Information science can be characterized as a mix of math, business discernment, devices, calculations and AI strategies, all of which assist us in figuring out the concealed experiences or examples from crude information which with canning be of significant use in the development of enormous business choices.

B. Information Scientist:

Information researchers inspect which questions need addressing and where to track down the connected information. They have business discernment and insightful abilities as well as the capacity to mine, clean, and present information. Organizations use information researchers to source, make due, and break down a lot of unstructured information. Required Skills for a Data Scientist: • Programming: Python, SQL, Scala, Java, MATLAB. • AI: Natural Language Processing, Classification, Clustering. • Information Visualization: Tableau, SAS, D3.js, Python, Java, R libraries. • Large information stages: MongoDB, Oracle, Microsoft Azure, Cloudera

2. SYSTEM ANALYSIS

2.1 EXISTING SYSTEM:

The death rate of under-five children in India declined last few decades, but few bigger states have poor performance. This is a matter of serious concern for the child's health as well as social development. Nowadays, machine learning techniques play a crucial role in the smart health care system to capture the hidden factors and patterns of outcomes.

In an existing system, the system used machine learning techniques to predict the important factors of under-five mortality. This study aims to explore the importance of machine learning techniques to predict under-five mortality and to find the important factors that cause under-five mortality. The data was taken from the National Family Health Survey-IV of Uttar Pradesh. We used four machine learning techniques like decision tree, support vector machine, random forest, and logistic regression to predict under-five mortality factors and model accuracy of each model. We have also used information gain to rank to know the important variables for accurate predictions in under-five mortality data.

2.2 DISADVANTAGES OF EXISTING SYSTEMS

• Data visualization doesn't provide an important set of tools for gaining qualitative insights.

• Data visualization and exploratory data analysis are not all fields, and he would recommend diving deeper into some of the books mentioned at the end.

2.3 PROPOSED SYSTEM:

The proposed model is to build a model to predict mortality. Collected data may contain missing values which may lead to inconsistencies. To get better results, the data should be preprocessed to improve the efficiency of the algorithm. Outliers should be removed and mutable conversions should also be performed. The data set collected to predict the given data is divided into training set and test set. In general, a ratio of 7:3 is applied to divide the training set and the test set. The data model created using machine learning algorithms is applied to the training set, and based on the accuracy of the test results, the prediction of the test set is made. The model can classify mortality. Different machine learning algorithms can be compared and the best algorithm can be used for classification

2.3 ADVANTAGES OF PROPOSED SYSTEM:

- The Naive Bayes algorithm is an intuitive method that uses the probability of each attribute belonging to each class to make predictions.
- Random forest or random decision forest is a synthetic learning method for classification, regression and other tasks, which works by building an infinite number of decision trees at the time of training and generating class as methods of classes (classification) or predictive mean (regression) of individual trees

3. SYSTEM STUDY

3.1 FEASIBILITY ANALYSIS

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

3.1.1 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.

3.1.2 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 for implementing this system.

3.1.3 SOCIAL FEASIBILTY

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 the system efficiently the user must not feel threatened by the system install 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.

4. HARDWARE AND SOFTWARE REQUIREMENTS

4.1 HARDWARE REQUIREMENTS

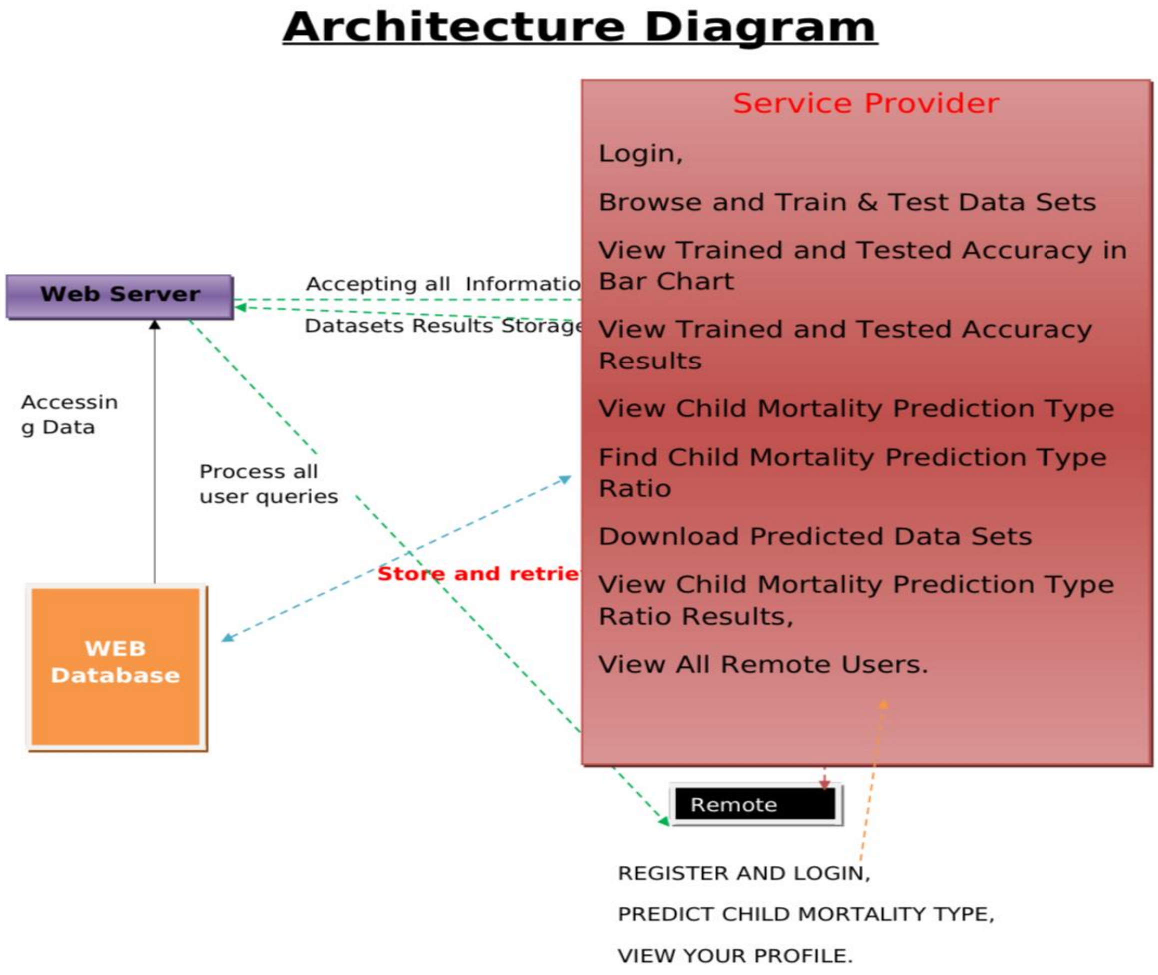
❖ **Processor**	-	Intel-i5
❖ **RAM**	-	4 GB (min)
❖ **Hard Disk**	-	20 GB
❖ **Key Board**	-	Standard Windows Keyboard
❖ **Mouse**	-	Two or Three Button Mouse
❖ **Monitor**	-	SVGA

4.2 SOFTWARE REQUIREMENTS

❖ **Operating system**	**:**	Windows 7 Ultimate.
❖ **Coding Language**	**:**	Python.
❖ **Front-End**	**:**	Python.
❖ **Back-End**	**:**	Django-ORM
❖ **Designing**	**:**	Html, css, javascript.
❖ **Data Base**	**:**	MySQL (WAMP Server).

5. ARCHITECTURE

5.1 ARCHITECTURE DIAGRAMS


Architecture Diagram
Service Provider
Login,
Browse and Train & Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Child Mortality Prediction Type
Find Child Mortality Prediction Type Ratio
Download Predicted Data Sets
View Child Mortality Prediction Type Ratio Results,
View All Remote Users.
Web Server
Accepting all Informatio
Datasets Results Storage
Accessin g Data
Process all user queries
Store and retrie
WEB Database
Remote
REGISTER AND LOGIN,
PREDICT CHILD MORTALITY TYPE,
VIEW YOUR PROFILE.

5.1 Architecture Diagrams

6. MODULES

Service Provider

In this module, the Service Provider has to login by using valid user name and password. After login successful he can do some operations such as Login, Browse and Train & Test Data Sets, View Trained and Tested Accuracy in Bar Chart, View Trained and Tested Accuracy Results, View Child Mortality Prediction Type, Find Child Mortality Prediction Type Ratio, Download Predicted Data Sets, View Child Mortality Prediction Type Ratio Results, View All Remote Users.

View and Authorize Users

In this module, the admin can view the list of users who all registered. In this, the admin can view the user’s details such as, user name, email, address and admin authorizes the users.

Remote User

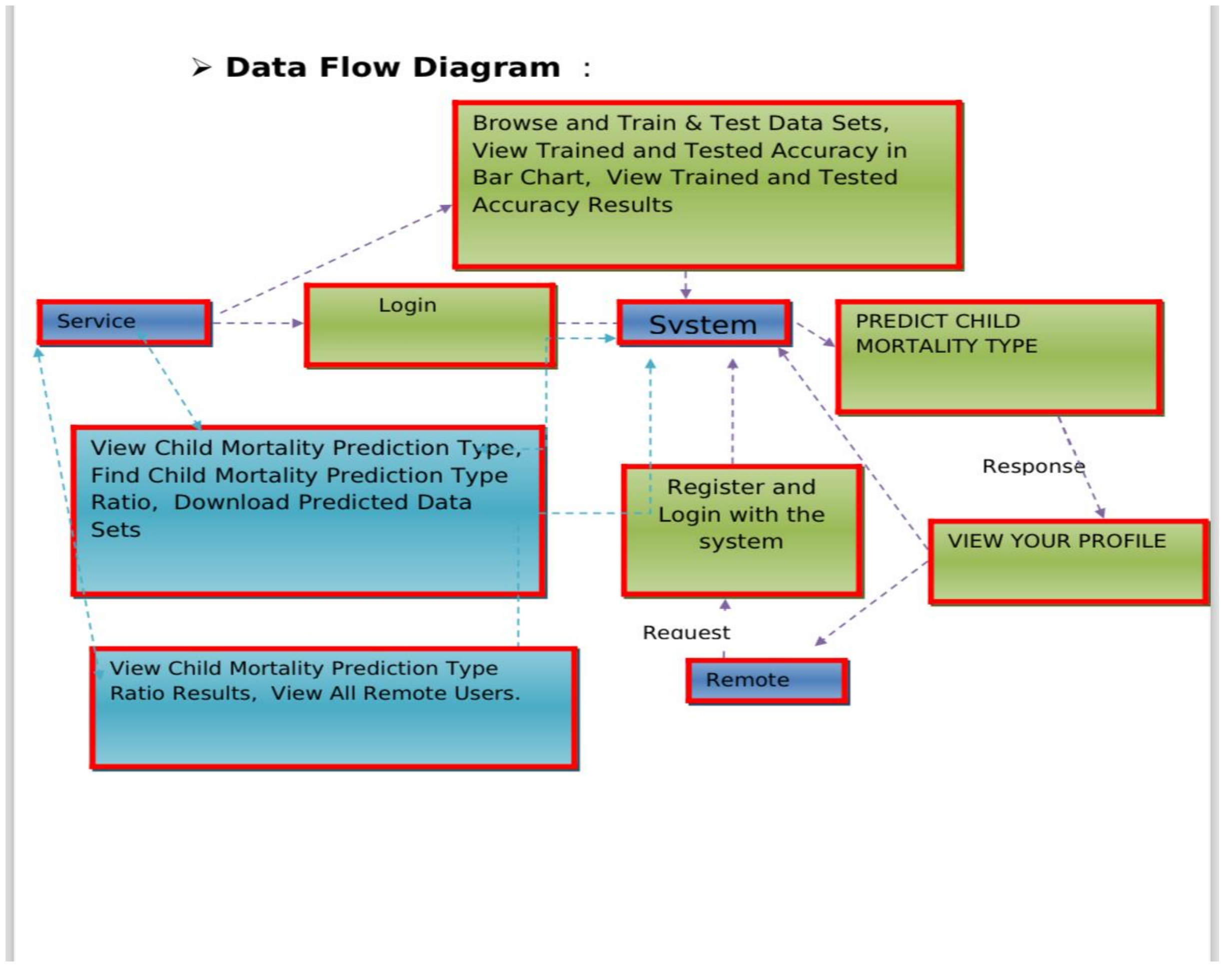
In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like REGISTER AND LOGIN, PREDICT RAINFALL ESTIMATE PREDICTION TYPE, VIEW YOUR PROFILE.

Model Execution:

Machine learning models such as Logistic Regression, Decision Tree Regression, Bayesian Ridge, Random Forest Regression and Gradient Boosting Regression we predict result. The MSE is appropriate for our regression problems since it is differentiable, contributing to the stability of the algorithms. It also heavily punishes the bigger errors over smaller errors. MAE is a risk providing metric which tells the expected value of the absolute error loss. Explained Variance Score proportion with which our machine learning model explains the scattering of the dataset is measured by this technique. R2 Score Goodness of fit is indicated by this metric and hence it measures the probability of the model to predict unknown samples, through the proportion of explained variance. The best score can be 1.0 and the score can also be negative.

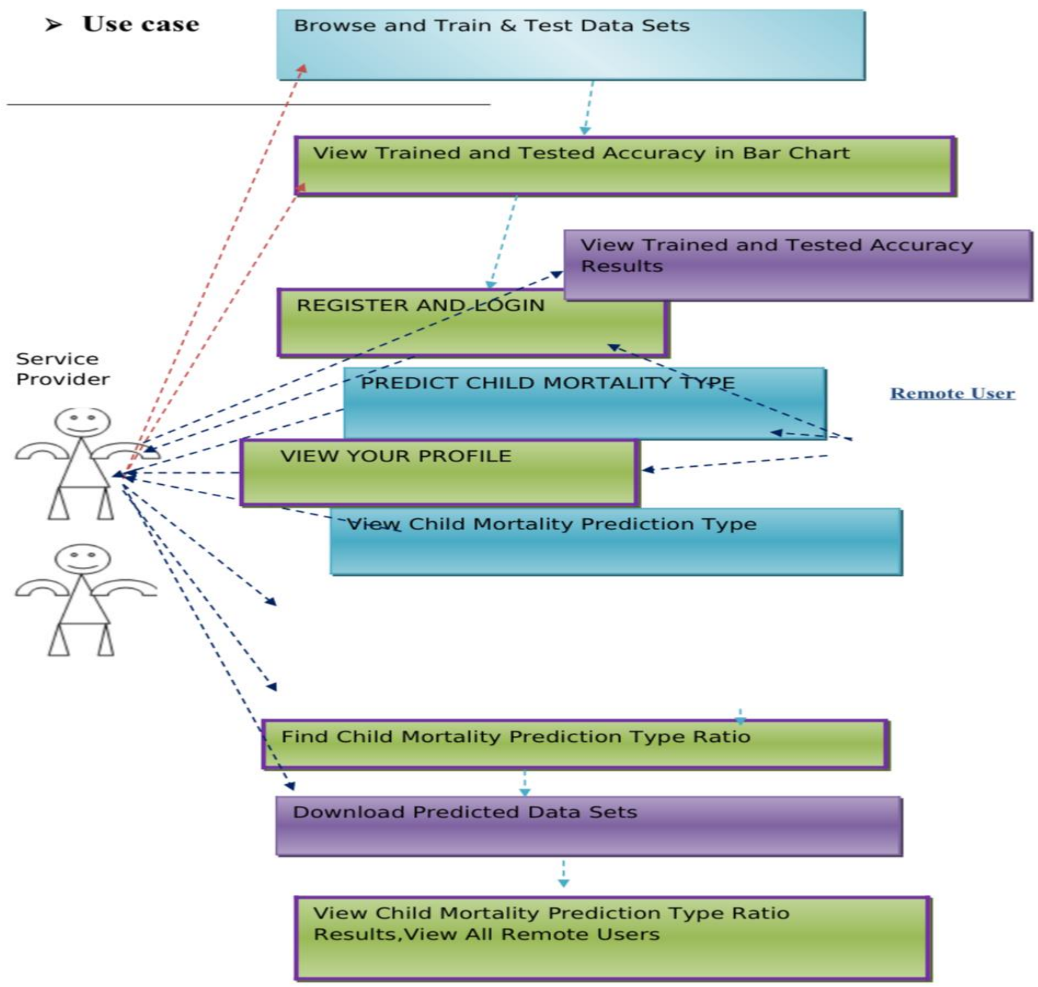
7. DIAGRAMS

7.1 DATA FLOW DIAGRAMS


➢ Data Flow Diagram :
Browse and Train & Test Data Sets, View Trained and Tested Accuracy in Bar Chart, View Trained and Tested Accuracy Results
Service
Login
Svstem
PREDICT CHILD MORTALITY TYPE
Response
View Child Mortality Prediction Type, Find Child Mortality Prediction Type Ratio, Download Predicted Data Sets
Register and Login with the system
VIEW YOUR PROFILE
Request
Remote
View Child Mortality Prediction Type Ratio Results, View All Remote Users.

7.1 Data Flow Diagrams

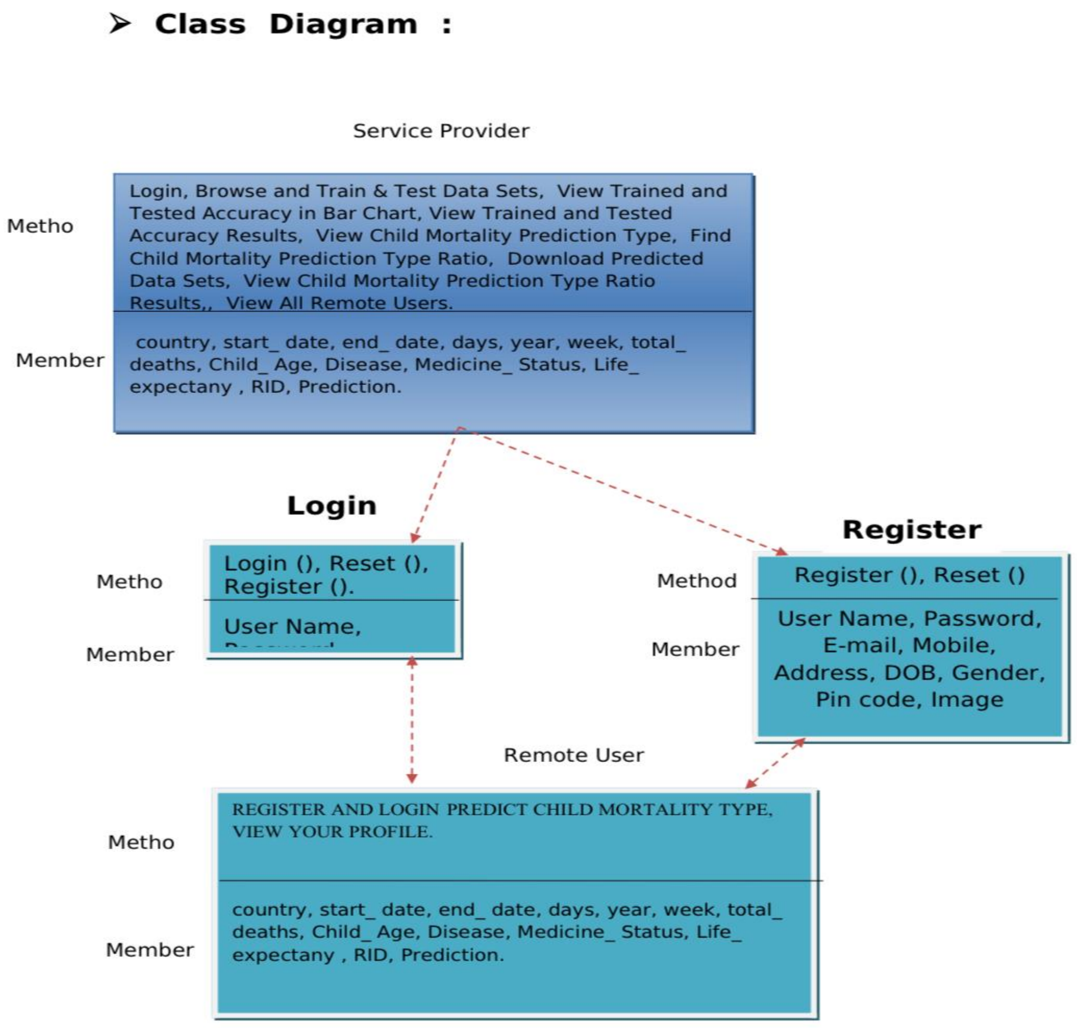
7.2 USE CASE DIAGRAM


➢ Use case
Browse and Train & Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
REGISTER AND LOGIN
Service Provider
PREDICT CHILD MORTALITY TYPE
Remote User
VIEW YOUR PROFILE
View Child Mortality Prediction Type
Find Child Mortality Prediction Type Ratio
Download Predicted Data Sets
View Child Mortality Prediction Type Ratio Results,View All Remote Users

7.2 Use-case Diagrams

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.

7.3 CLASS DIAGRAM

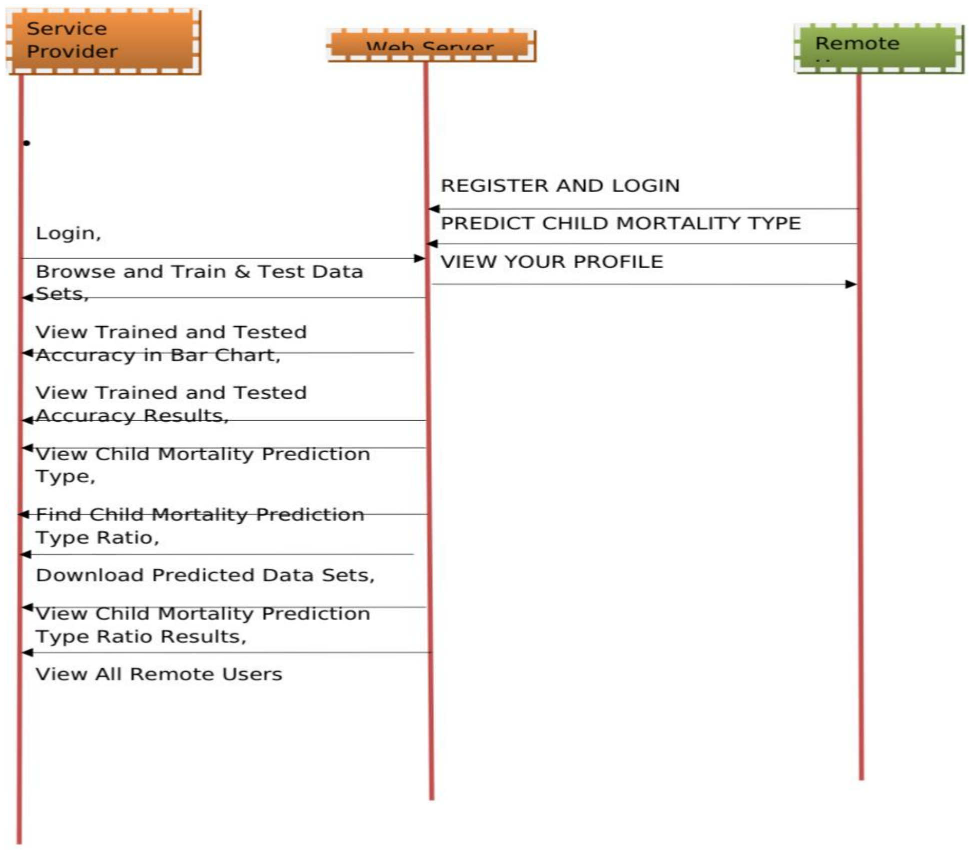

➤ Class Diagram :
Service Provider
Metho
Login, Browse and Train & Test Data Sets, View Trained and Tested Accuracy in Bar Chart, View Trained and Tested Accuracy Results, View Child Mortality Prediction Type, Find Child Mortality Prediction Type Ratio, Download Predicted Data Sets, View Child Mortality Prediction Type Ratio Results,, View All Remote Users.
Member
country, start_ date, end_ date, days, year, week, total_ deaths, Child_ Age, Disease, Medicine_ Status, Life_ expectany , RID, Prediction.
Login
Metho
Login (), Reset (), Register ().
Member
User Name,
Register
Method
Register (), Reset ()
Member
User Name, Password, E-mail, Mobile, Address, DOB, Gender, Pin code, Image
Remote User
Metho
REGISTER AND LOGIN PREDICT CHILD MORTALITY TYPE, VIEW YOUR PROFILE.
Member
country, start_ date, end_ date, days, year, week, total_ deaths, Child_ Age, Disease, Medicine_ Status, Life_ expectany , RID, Prediction.

7.3 Class Diagrams

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.

7.4 SEQUENCE DIAGRAM

➢ **Sequence Diagram**



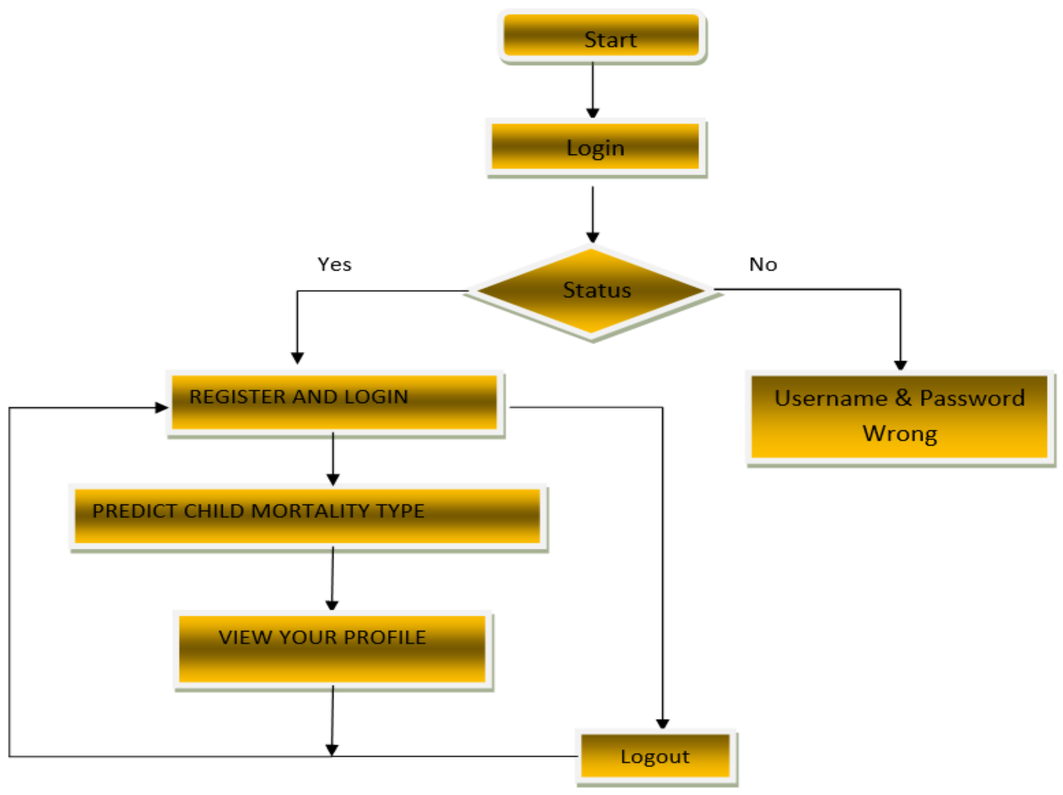
Service Provider
Web Server
Remote
REGISTER AND LOGIN
PREDICT CHILD MORTALITY TYPE
VIEW YOUR PROFILE
Login,
Browse and Train & Test Data Sets,
View Trained and Tested Accuracy in Bar Chart,
View Trained and Tested Accuracy Results,
View Child Mortality Prediction Type,
Find Child Mortality Prediction Type Ratio,
Download Predicted Data Sets,
View Child Mortality Prediction Type Ratio Results,
View All Remote Users

7.4 Sequence Diagrams

Sequence diagrams model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes. Sequence diagrams are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system. Other dynamic modeling techniques include activity diagramming, communication diagramming, timing diagramming, and interaction overview.

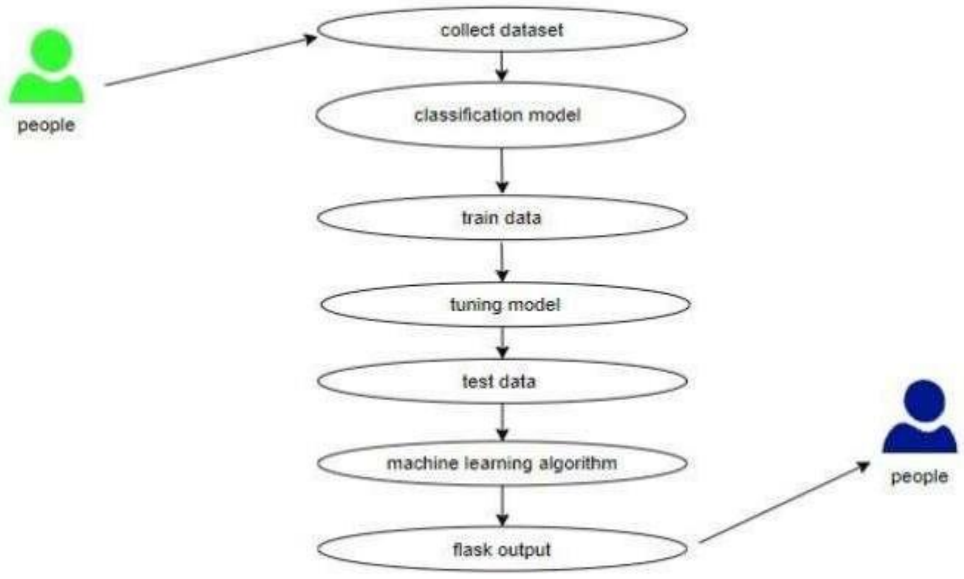
7.5 FLOW CHART DIAGRAMS

➢ **Flow Chart : Remote User**



7.5 Flow chart Diagram

7.6 COMPONENT DIAGRAMS



7.6 Component Diagrams

Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analyzed the functionalities are captured in use cases. So, it can say that uses cases are nothing but the system functionalities written in an organized manner

8. IMPLEMENTATION

8.1 SOURCE CODE

8.1.1 User side - views.py

```
from django.db.models import Count
from django.db.models import Q
from django.shortcuts import render, redirect, get_object_or_404
import numpy as np # linear algebra
import pandas as pd
from sklearn.ensemble import VotingClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.metrics import accuracy_score

# Create your views here.
from Remote_User.models import ClientRegister_Model,child_mortality_type,detection_ratio,detection_accuracy

def login(request):


    if request.method == "POST" and 'submit1' in request.POST:

        username = request.POST.get('username')
        password = request.POST.get('password')
        try:
            enter = ClientRegister_Model.objects.get(username=username,password=password)
            request.session["userid"] = enter.id
            return redirect('ViewYourProfile')
        except:
            pass

    return render(request,'RUser/login.html')

def Register1(request):
    if request.method == "POST":
        username = request.POST.get('username')
        email = request.POST.get('email')
        password = request.POST.get('password')
        phoneno = request.POST.get('phoneno')
        country = request.POST.get('country')
        state = request.POST.get('state')
        city = request.POST.get('city')
        address = request.POST.get('address')
        gender = request.POST.get('gender')
        ClientRegister_Model.objects.create(username=username, email=email, password=password,
phoneno=phoneno,
                                            country=country, state=state, city=city, address=address, gender=gender)
        obj = "Registered Successfully"
        return render(request, 'RUser/Register1.html', {'object': obj})
    else:
        return render(request,'RUser/Register1.html')

def ViewYourProfile(request):
    userid = request.session['userid']
    obj = ClientRegister_Model.objects.get(id= userid)
    return render(request,'RUser/ViewYourProfile.html',{'object':obj})



def Child_Mortality_prediction_Type(request):
```

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

    country= request.POST.get('country')
    start_date= request.POST.get('start_date')
    end_date= request.POST.get('end_date')
    days= request.POST.get('days')
    year= request.POST.get('year')
    week= request.POST.get('week')
    total_deaths= request.POST.get('total_deaths')
    Child_Age= request.POST.get('Child_Age')
    Disease= request.POST.get('Disease')
    Medicine_Status= request.POST.get('Medicine_Status')
    Life_expectancy= request.POST.get('Life_expectancy')
    RID= request.POST.get('RID')




    df = pd.read_csv('Healthcare_Datasets.csv')
    df
    df.columns

    def apply_results(results):
        if (results == 'No'):
            return 0
        elif (results == 'Yes'):
            return 1

    df['Results'] = df['Label'].apply(apply_results)

    X = df['RID']
    y = df['Results']
```

```
print("RID")
print(X)
print("Results")
print(y)

cv = CountVectorizer(lowercase=False, strip_accents='unicode', ngram_range=(1, 1))
#X = cv.fit_transform(df['RID'].apply(lambda x: np.str_(X)))
X = cv.fit_transform(X)

models = []
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
X_train.shape, X_test.shape, y_train.shape

print("Naive Bayes")

from sklearn.naive_bayes import MultinomialNB
NB = MultinomialNB()
NB.fit(X_train, y_train)
predict_nb = NB.predict(X_test)
naivebayes = accuracy_score(y_test, predict_nb) * 100
print(naivebayes)
print(confusion_matrix(y_test, predict_nb))
print(classification_report(y_test, predict_nb))
models.append(('naive_bayes', NB))


# SVM Model
print("SVM")
from sklearn import svm
lin_clf = svm.LinearSVC()
```

```
lin_clf.fit(X_train, y_train)
predict_svm = lin_clf.predict(X_test)
svm_acc = accuracy_score(y_test, predict_svm) * 100
print(svm_acc)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, predict_svm))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, predict_svm))
models.append(('svm', lin_clf))
detection_accuracy.objects.create(names="SVM", ratio=svm_acc)

print("Logistic Regression")

from sklearn.linear_model import LogisticRegression
reg = LogisticRegression(random_state=0, solver='lbfgs').fit(X_train, y_train)
y_pred = reg.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, y_pred) * 100)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, y_pred))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, y_pred))
models.append(('logistic', reg))


print("KNeighborsClassifier")

from sklearn.neighbors import KNeighborsClassifier
kn = KNeighborsClassifier()
kn.fit(X_train, y_train)
knpredict = kn.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, knpredict) * 100)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, knpredict))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, knpredict))



print("Random Forest Classifier")
from sklearn.ensemble import RandomForestClassifier
rf_clf = RandomForestClassifier()
rf_clf.fit(X_train, y_train)
rfpredict = rf_clf.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, rfpredict) * 100)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, rfpredict))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, rfpredict))
models.append(('RandomForestClassifier', rf_clf))

classifier = VotingClassifier(models)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

RID = [RID]
vector1 = cv.transform(RID).toarray()
predict_text = classifier.predict(vector1)

pred = str(predict_text).replace("[", "")
pred1 = pred.replace("]", "")

prediction = int(pred1)
```

```
    if prediction == 0:
        val = 'Low Death Ratio'
    else:
        val = 'High Death Ratio'

    print(val)
    print(pred1)

    child_mortality_type.objects.create(country=country,
    start_date=start_date,
    end_date=end_date,
    days=days,
    year=year,
    week=week,
    total_deaths=total_deaths,
    Child_Age=Child_Age,
    Disease=Disease,
    Medicine_Status=Medicine_Status,
    Life_expectancy=Life_expectancy,
    RID=RID,
    Prediction=val)

    return render(request, 'RUser/Child_Mortality_prediction_Type.html',{'objs': val})
return render(request, 'RUser/Child_Mortality_prediction_Type.html')
```

8.1.2 User Side- modles.py

```
from django.db import models

# Create your models here.
from django.db.models import CASCADE



class ClientRegister_Model(models.Model):

    username = models.CharField(max_length=30)
    email = models.EmailField(max_length=30)
    password = models.CharField(max_length=10)
    phoneno = models.CharField(max_length=10)
    country = models.CharField(max_length=30)
    state = models.CharField(max_length=30)
    city = models.CharField(max_length=30)
    address= models.CharField(max_length=300)
    gender= models.CharField(max_length=30)

class child_mortality_type(models.Model):



    country= models.CharField(max_length=3000)
    start_date= models.CharField(max_length=3000)
    end_date= models.CharField(max_length=3000)
    days= models.CharField(max_length=3000)
    year= models.CharField(max_length=3000)
    week= models.CharField(max_length=3000)
    total_deaths= models.CharField(max_length=3000)
    Child_Age= models.CharField(max_length=3000)
    Disease= models.CharField(max_length=3000)
    Medicine_Status= models.CharField(max_length=3000)
    Life_expectancy= models.CharField(max_length=3000)
    RID= models.CharField(max_length=3000)
    Prediction= models.CharField(max_length=3000)


class detection_accuracy(models.Model):

    names = models.CharField(max_length=300)
    ratio = models.CharField(max_length=300)

class detection_ratio(models.Model):

    names = models.CharField(max_length=300)
    ratio = models.CharField(max_length=300)
```

8.1.3 User side- forms.py:

```
from django import forms

from Remote_User.models import ClientRegister_Model


class ClientRegister_Form(forms.ModelForm):
    password = forms.CharField(widget=forms.PasswordInput())
    email = forms.EmailField(required=True)

    class Meta:
        model = ClientRegister_Model
        fields = ("username","email","password","phoneno","country","state","city")
```

8.1.4 Service provider side- views.py:

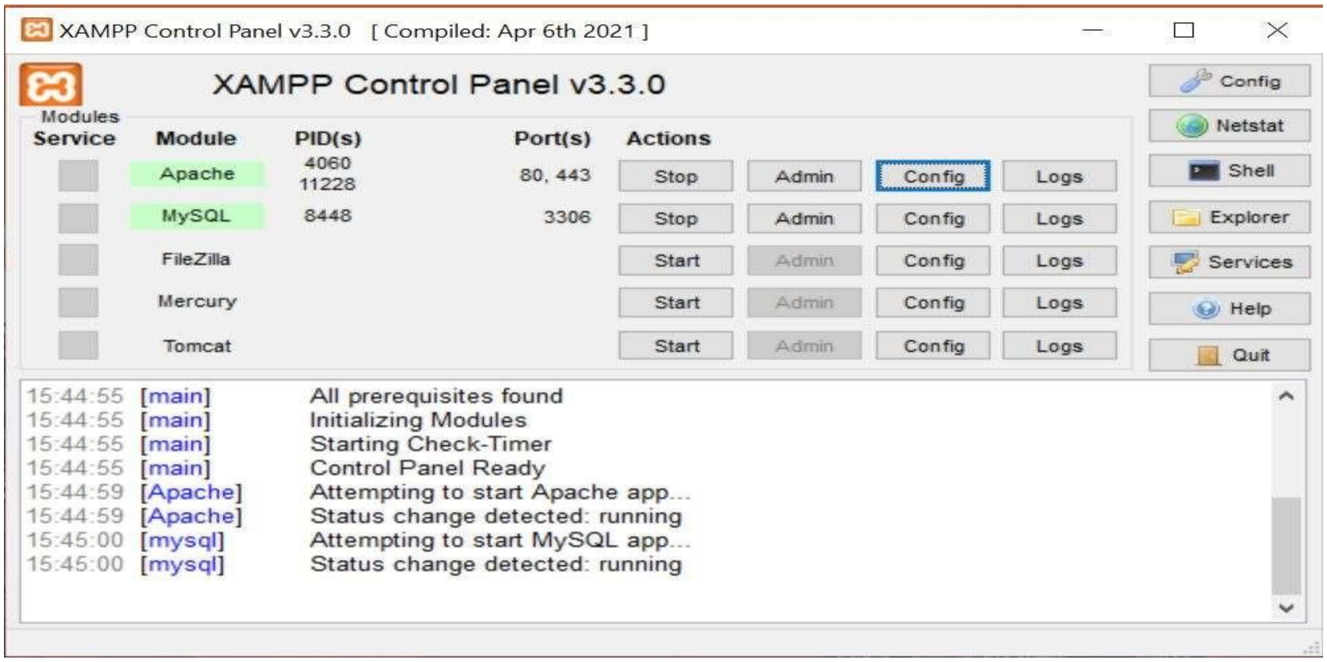
```
from django.db.models import  Count, Avg
from django.shortcuts import render, redirect
from django.db.models import Count
from django.db.models import Q
import datetime
import xlwt
from django.http import HttpResponse
import numpy as np # linear algebra
import pandas as pd

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier

# Create your views here.
from Remote_User.models import ClientRegister_Model,child_mortality_type,detection_ratio,detection_accuracy


def serviceproviderlogin(request):
    if request.method  == "POST":
        admin = request.POST.get('username')
        password = request.POST.get('password')
        if admin == "Admin" and password =="Admin":
            return redirect('View_Remote_Users')

    return render(request,'SProvider/serviceproviderlogin.html')

def View_Child_Mortality_Prediction_Type_Ratio(request):
    detection_ratio.objects.all().delete()
    ratio = ""
    kword = 'High Death Ratio'
    print(kword)
    obj = child_mortality_type.objects.all().filter(Q(Prediction=kword))
    obj1 = child_mortality_type.objects.all()
    count = obj.count();
    count1 = obj1.count();
    ratio = (count / count1) * 100
    if ratio != 0:
        detection_ratio.objects.create(names=kword, ratio=ratio)

    ratio1 = ""
    kword1 = 'Low Death Ratio'
    print(kword1)
    obj1 = child_mortality_type.objects.all().filter(Q(Prediction=kword1))
    obj11 = child_mortality_type.objects.all()
    count1 = obj1.count();
    count11 = obj11.count();
    ratio1 = (count1 / count11) * 100
    if ratio1 != 0:
        detection_ratio.objects.create(names=kword1, ratio=ratio1)

    obj = detection_ratio.objects.all()
    return render(request, 'SProvider/View_Child_Mortality_Prediction_Type_Ratio.html', {'objs': obj})

def View_Remote_Users(request):
    obj=ClientRegister_Model.objects.all()
    return render(request,'SProvider/View_Remote_Users.html',{'objects':obj})

def ViewTrendings(request):
    topic = child_mortality_type.objects.values('topics').annotate(dcount=Count('topics')).order_by('-dcount')
    return  render(request,'SProvider/ViewTrendings.html',{'objects':topic})

def charts(request,chart_type):
    chart1 = detection_ratio.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts.html", {'form':chart1, 'chart_type':chart_type})

def charts1(request,chart_type):
    chart1 = detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts1.html", {'form':chart1, 'chart_type':chart_type})

def View_Child_Mortality_Prediction_Type(request):
    obj =child_mortality_type.objects.all()
    return render(request, 'SProvider/View_Child_Mortality_Prediction_Type.html', {'list_objects': obj})

def likeschart(request,like_chart):
    charts =detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/likeschart.html", {'form':charts, 'like_chart':like_chart})


def Download_Trained_DataSets(request):

    response = HttpResponse(content_type='application/ms-excel')
    # decide file name
    response['Content-Disposition'] = 'attachment; filename="PredictedData.xls"'
    # creating workbook
    wb = xlwt.Workbook(encoding='utf-8')
    # adding sheet
    ws = wb.add_sheet("sheet1")
    # Sheet header, first row
    row_num = 0
    font_style = xlwt.XFStyle()
    # headers are bold
    font_style.font.bold = True
    # writer = csv.writer(response)
    obj = child_mortality_type.objects.all()
    data = obj  # dummy method to fetch data.
    for my_row in data:
        row_num = row_num + 1

        ws.write(row_num, 0, my_row.country, font_style)
        ws.write(row_num, 1, my_row.start_date, font_style)
        ws.write(row_num, 2, my_row.end_date, font_style)
        ws.write(row_num, 3, my_row.days, font_style)
        ws.write(row_num, 4, my_row.year, font_style)
        ws.write(row_num, 5, my_row.week, font_style)
        ws.write(row_num, 6, my_row.total_deaths, font_style)
        ws.write(row_num, 7, my_row.Child_Age, font_style)
        ws.write(row_num, 8, my_row.Disease, font_style)
        ws.write(row_num, 9, my_row.Medicine_Status, font_style)
        ws.write(row_num, 10, my_row.Life_expectancy, font_style)
        ws.write(row_num, 11, my_row.RID, font_style)
        ws.write(row_num, 12, my_row.Prediction, font_style)

    wb.save(response)
    return response

def Train_Test_DataSets(request):
    detection_accuracy.objects.all().delete()

    df = pd.read_csv('Healthcare_Datasets.csv')
    df
    df.columns

    def apply_results(results):
        if (results == 'No'):
            return 0
        elif (results == 'Yes'):
            return 1

    df['Results'] = df['Label'].apply(apply_results)

    X = df['RID']
    y = df['Results']

    print("RID")
    print(X)
    print("Results")
    print(y)

    cv = CountVectorizer(lowercase=False, strip_accents='unicode', ngram_range=(1, 1))
    # X = cv.fit_transform(df['RID'].apply(lambda x: np.str_(X)))
    X = cv.fit_transform(X)

    models = []
    from sklearn.model_selection import train_test_split
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
```

```
X_train.shape, X_test.shape, y_train.shape

print("Naive Bayes")

from sklearn.naive_bayes import MultinomialNB
NB = MultinomialNB()
NB.fit(X_train, y_train)
predict_nb = NB.predict(X_test)
naivebayes = accuracy_score(y_test, predict_nb) * 100
print(naivebayes)
print(confusion_matrix(y_test, predict_nb))
print(classification_report(y_test, predict_nb))
models.append(('naive_bayes', NB))
detection_accuracy.objects.create(names="Naive Bayes", ratio=naivebayes)

# SVM Model
print("SVM")
from sklearn import svm
lin_clf = svm.LinearSVC()
lin_clf.fit(X_train, y_train)
predict_svm = lin_clf.predict(X_test)
svm_acc = accuracy_score(y_test, predict_svm) * 100
print(svm_acc)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, predict_svm))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, predict_svm))
models.append(('svm', lin_clf))
detection_accuracy.objects.create(names="SVM", ratio=svm_acc)


print("Logistic Regression")

from sklearn.linear_model import LogisticRegression
reg = LogisticRegression(random_state=0, solver='lbfgs').fit(X_train, y_train)
y_pred = reg.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, y_pred) * 100)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, y_pred))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, y_pred))
models.append(('logistic', reg))

detection_accuracy.objects.create(names="Logistic Regression", ratio=accuracy_score(y_test, y_pred) * 100)


print("KNeighborsClassifier")

from sklearn.neighbors import KNeighborsClassifier
kn = KNeighborsClassifier()
```

```
    kn.fit(X_train, y_train)
    knpredict = kn.predict(X_test)
    print("ACCURACY")
    print(accuracy_score(y_test, knpredict) * 100)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, knpredict))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, knpredict))
    detection_accuracy.objects.create(names="KNeighborsClassifier", ratio=accuracy_score(y_test, knpredict) * 100)

    print("Random Forest Classifier")
    from sklearn.ensemble import RandomForestClassifier
    rf_clf = RandomForestClassifier()
    rf_clf.fit(X_train, y_train)
    rfpredict = rf_clf.predict(X_test)
    print("ACCURACY")
    print(accuracy_score(y_test, rfpredict) * 100)
    print("CLASSIFICATION REPORT")
    print(classification_report(y_test, rfpredict))
    print("CONFUSION MATRIX")
    print(confusion_matrix(y_test, rfpredict))
    models.append(('RandomForestClassifier', rf_clf))
    detection_accuracy.objects.create(names="Random Forest Classifier", ratio=accuracy_score(y_test, rfpredict) *
100)


    predicts = 'Results.csv'
    df.to_csv(predicts, index=False)
    df.to_markdown

    obj = detection_accuracy.objects.all()


    return render(request,'SProvider/Train_Test_DataSets.html', {'objs': obj})
```

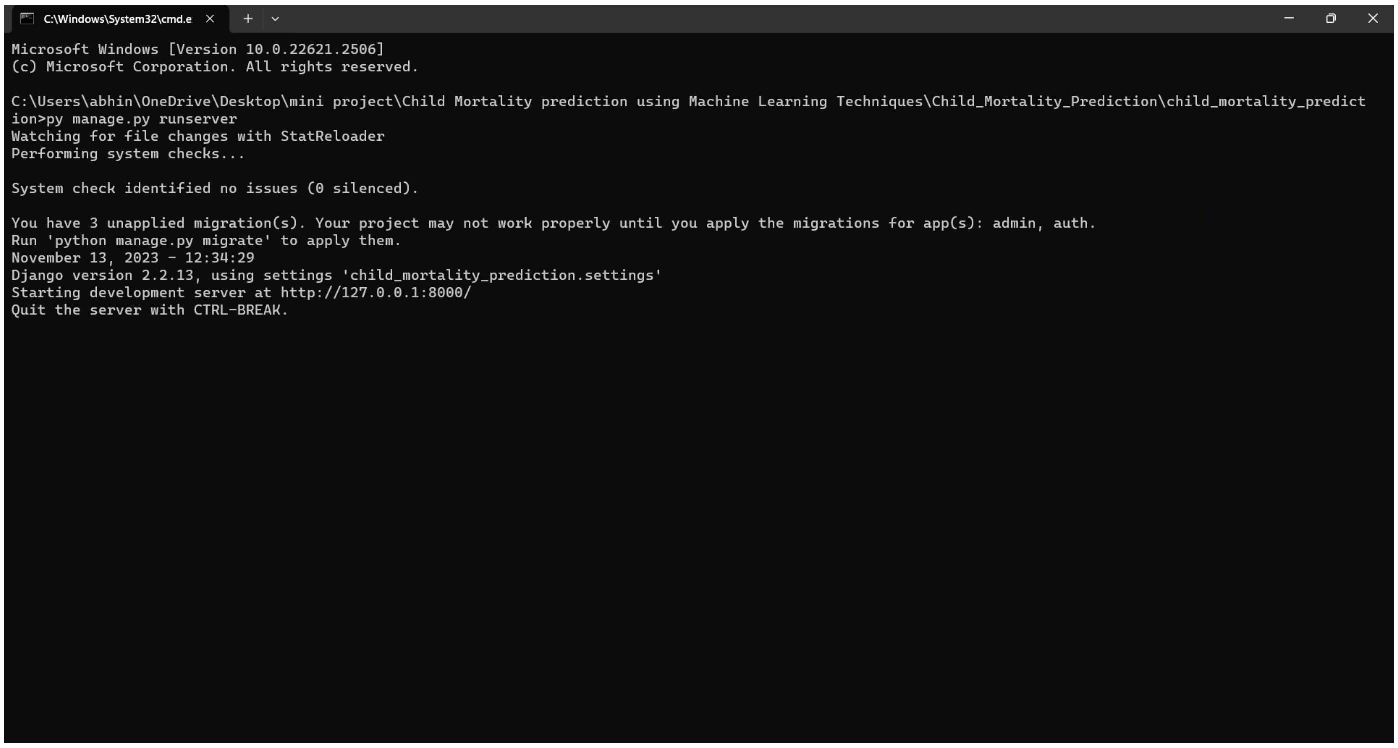
9. SCREENSHOTS

9.1 Input:


XAMPP Control Panel v3.3.0 [Compiled: Apr 6th 2021]
XAMPP Control Panel v3.3.0
Config
Netstat
Shell
Explorer
Services
Help
Quit

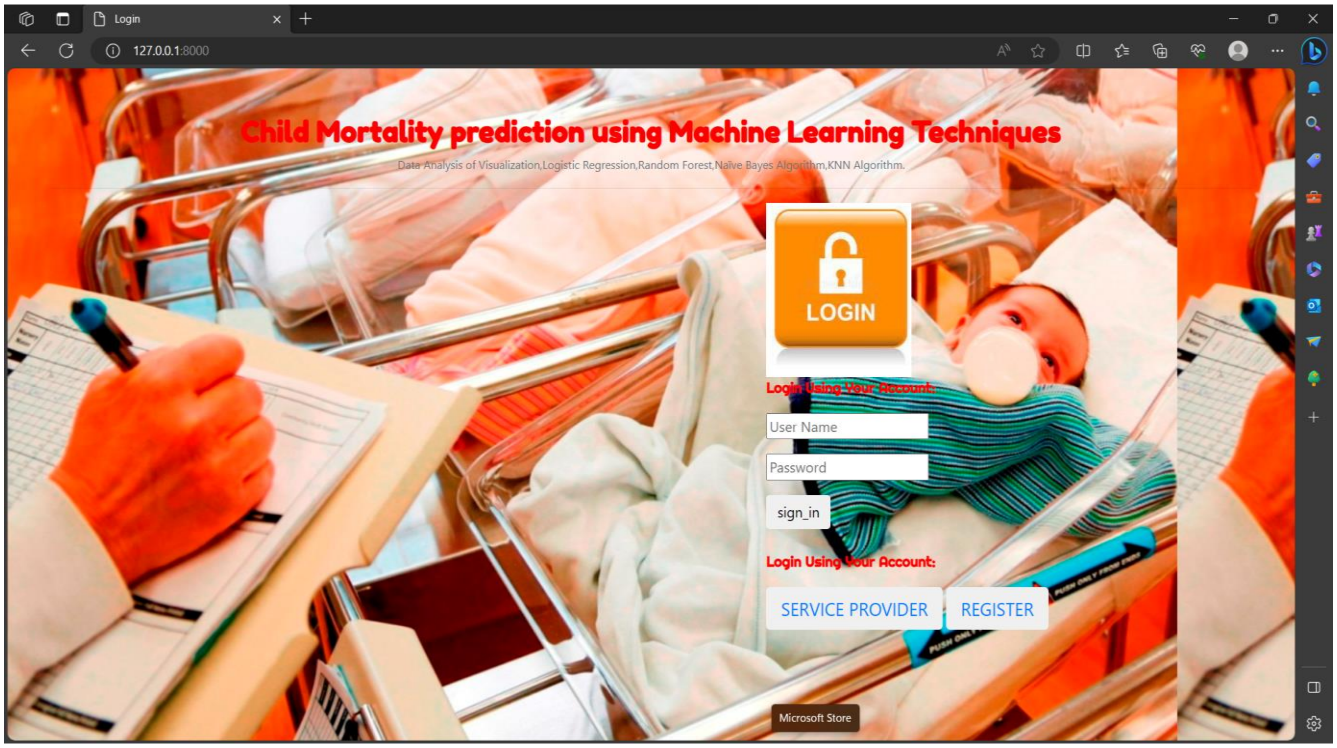
```
Modules
Service  Module     PID(s)   Port(s)   Actions
         Apache     4060     80, 443   Stop   Admin  Config  Logs
                    11228
         MySQL      8448     3306      Stop   Admin  Config  Logs
         FileZilla                     Start  Admin  Config  Logs
         Mercury                       Start  Admin  Config  Logs
         Tomcat                        Start  Admin  Config  Logs

15:44:55  [main]      All prerequisites found
15:44:55  [main]      Initializing Modules
15:44:55  [main]      Starting Check-Timer
15:44:55  [main]      Control Panel Ready
15:44:59  [Apache]    Attempting to start Apache app...
15:44:59  [Apache]    Status change detected: running
15:45:00  [mysql]     Attempting to start MySQL app...
15:45:00  [mysql]     Status change detected: running
```

9.1 XAMPP Server

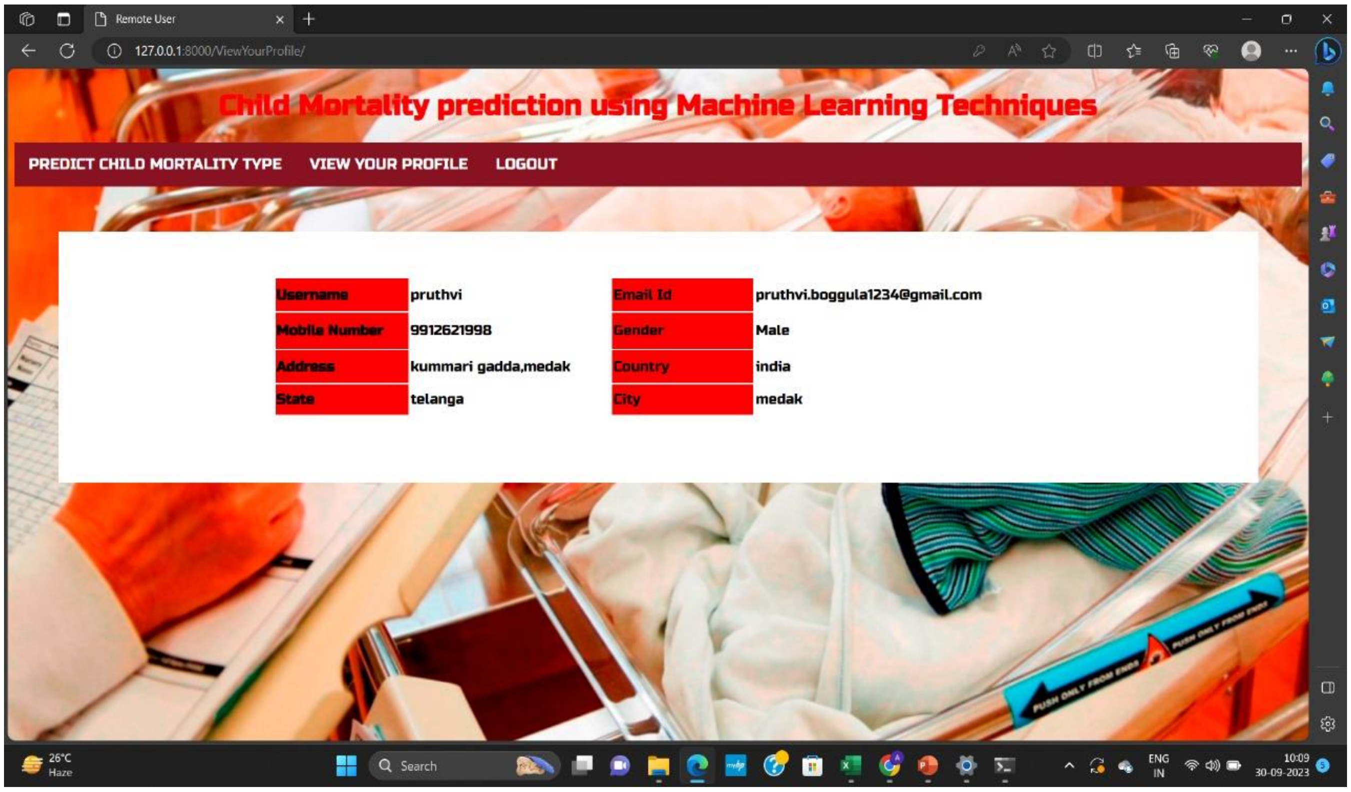


```
Microsoft Windows [Version 10.0.22621.2506]
(c) Microsoft Corporation. All rights reserved.

C:\Users\abhin\OneDrive\Desktop\mini project\Child Mortality prediction using Machine Learning Techniques\Child_Mortality_Prediction\child_mortality_predict
ion>py manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 3 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth.
Run 'python manage.py migrate' to apply them.
November 13, 2023 - 12:34:29
Django version 2.2.13, using settings 'child_mortality_prediction.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

9.2 Command Prompt

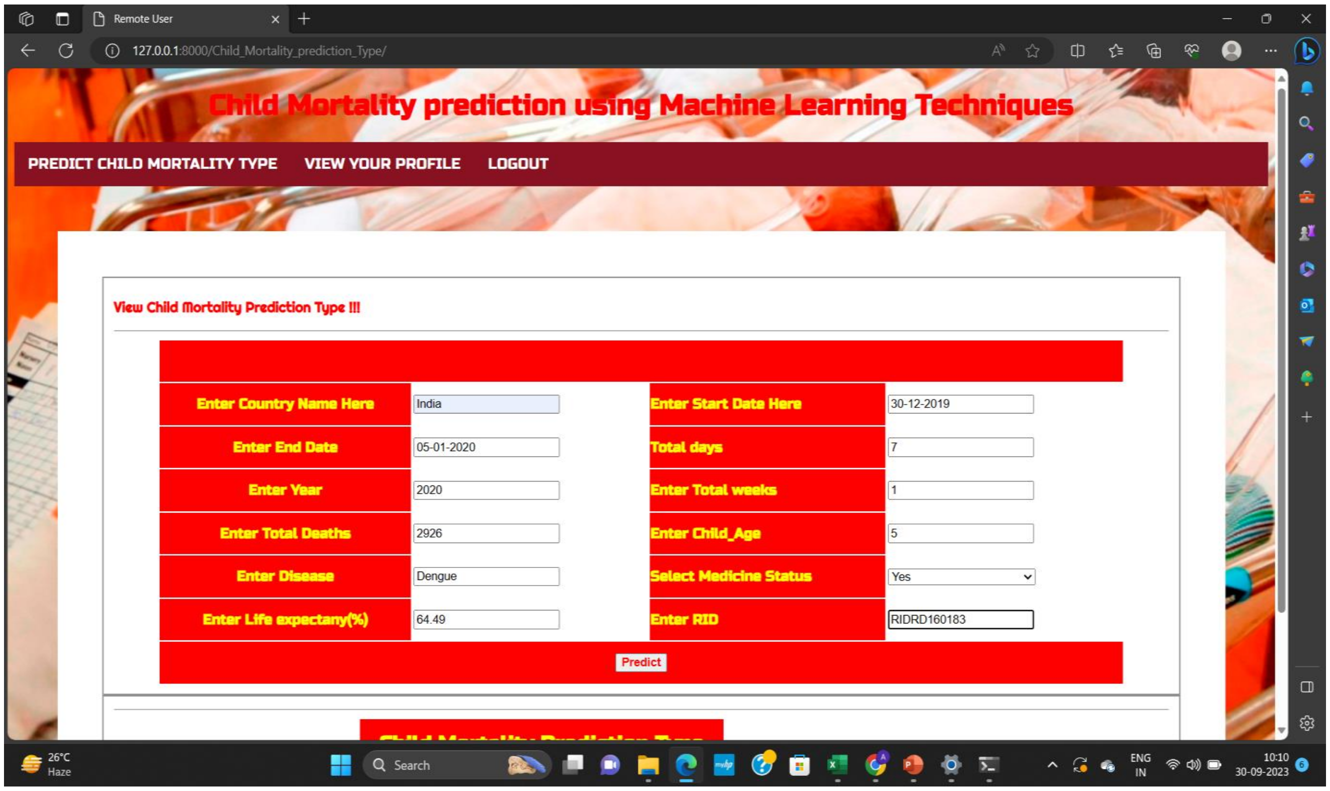

Login
127.0.0.1:8000
Child Mortality prediction using Machine Learning Techniques
Data Analysis of Visualization,Logistic Regression,Random Forest,Naive Bayes Algorithm,KNN Algorithm.
LOGIN
Login Using Your Account:
User Name
Password
sign_in
Login Using Your Account:
SERVICE PROVIDER
REGISTER
Microsoft Store

9.3 Login Page


Remote User
127.0.0.1:8000/ViewYourProfile/
Child Mortality prediction using Machine Learning Techniques
PREDICT CHILD MORTALITY TYPE
VIEW YOUR PROFILE
LOGOUT

Username	pruthvi	Email Id	pruthvi.boggula1234@gmail.com
Mobile Number	9912621998	Gender	Male
Address	kummari gadda,medak	Country	india
State	telanga	City	medak

9.4 User Registration Page


Child Mortality prediction using Machine Learning Techniques

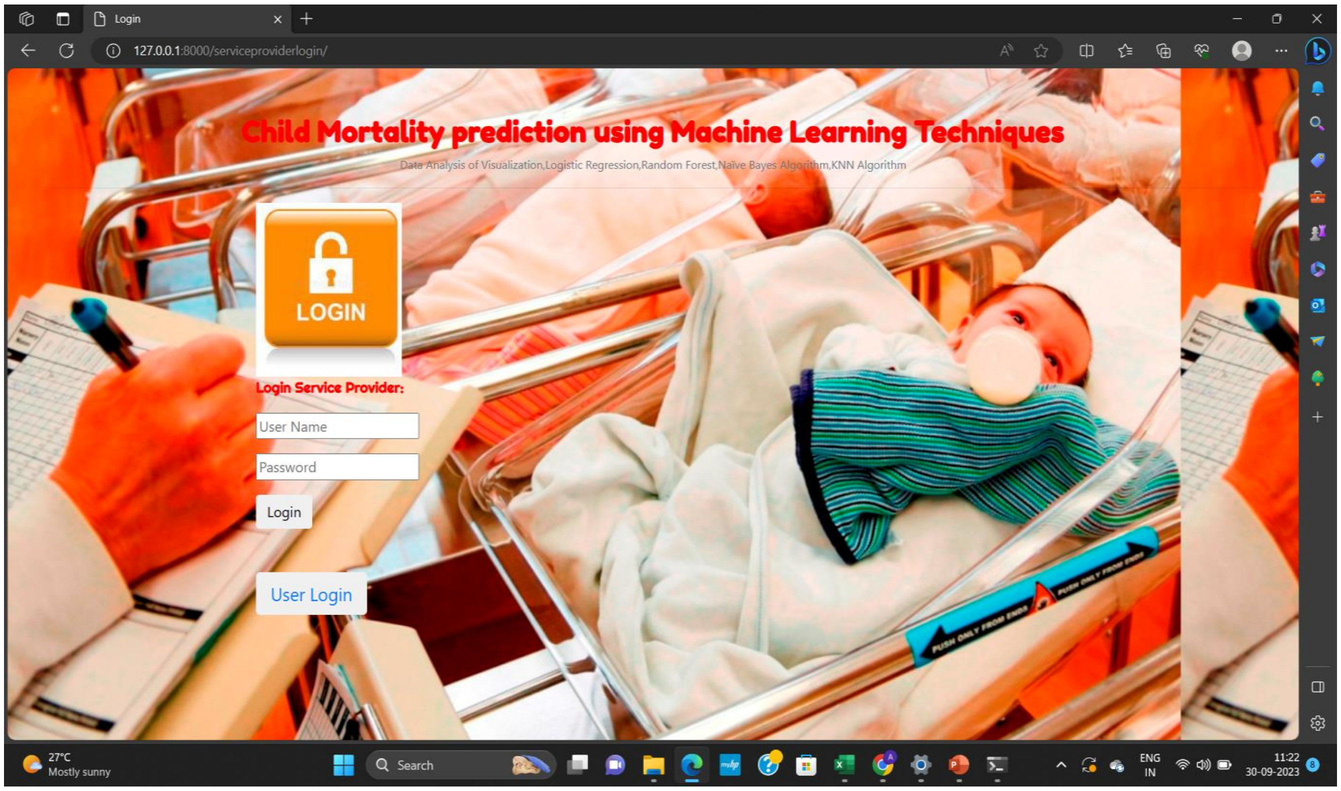
PREDICT CHILD MORTALITY TYPE VIEW YOUR PROFILE LOGOUT

View Child Mortality Prediction Type !!!

Enter Country Name Here	India	Enter Start Date Here	30-12-2019
Enter End Date	05-01-2020	Total days	7
Enter Year	2020	Enter Total weeks	1
Enter Total Deaths	2926	Enter Child_Age	5
Enter Disease	Dengue	Select Medicine Status	Yes
Enter Life expectany(%)	64.49	Enter RID	RIDRD160183

Predict

9.5 Data Input


Child Mortality prediction using Machine Learning Techniques

Data Analysis of Visualization,Logistic Regression,Random Forest,Naive Bayes Algorithm,KNN Algorithm

Login Service Provider:

User Name

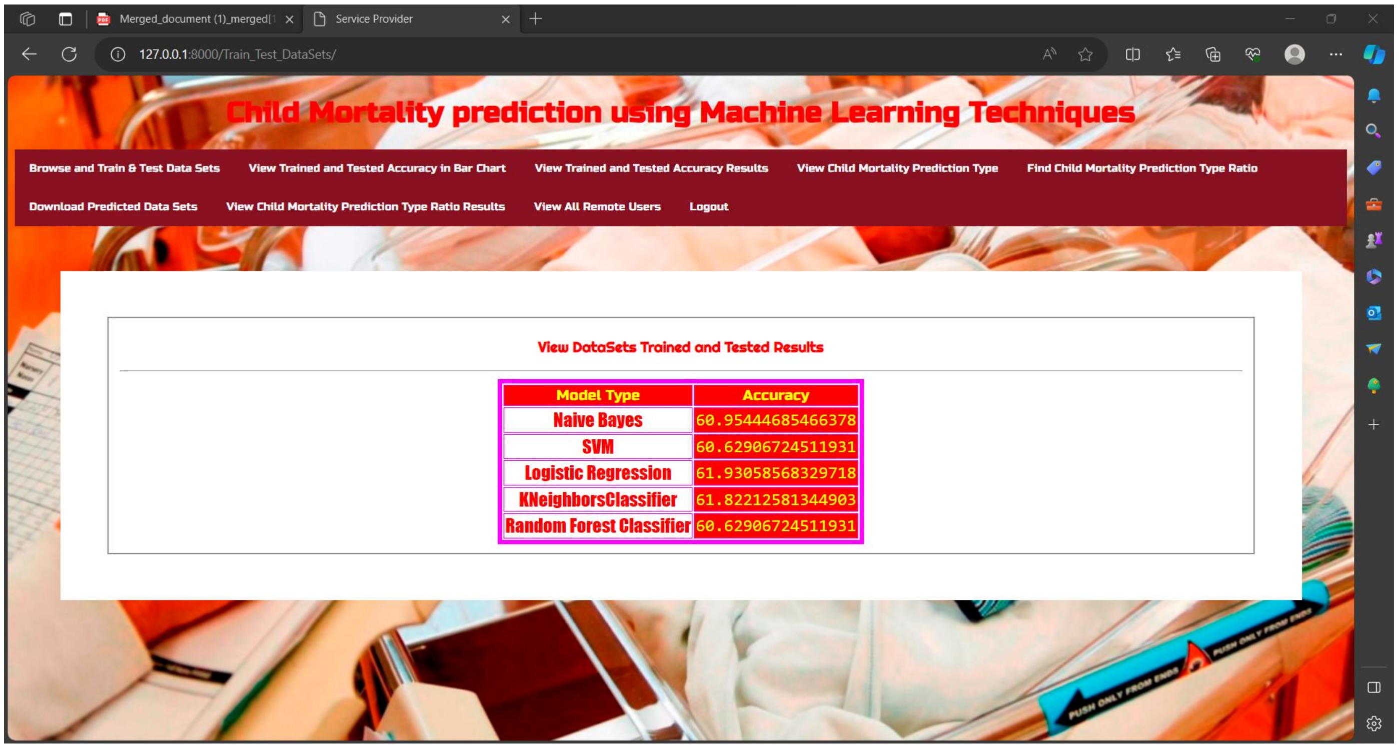
Password

Login

User Login

9.6 Admin Login page

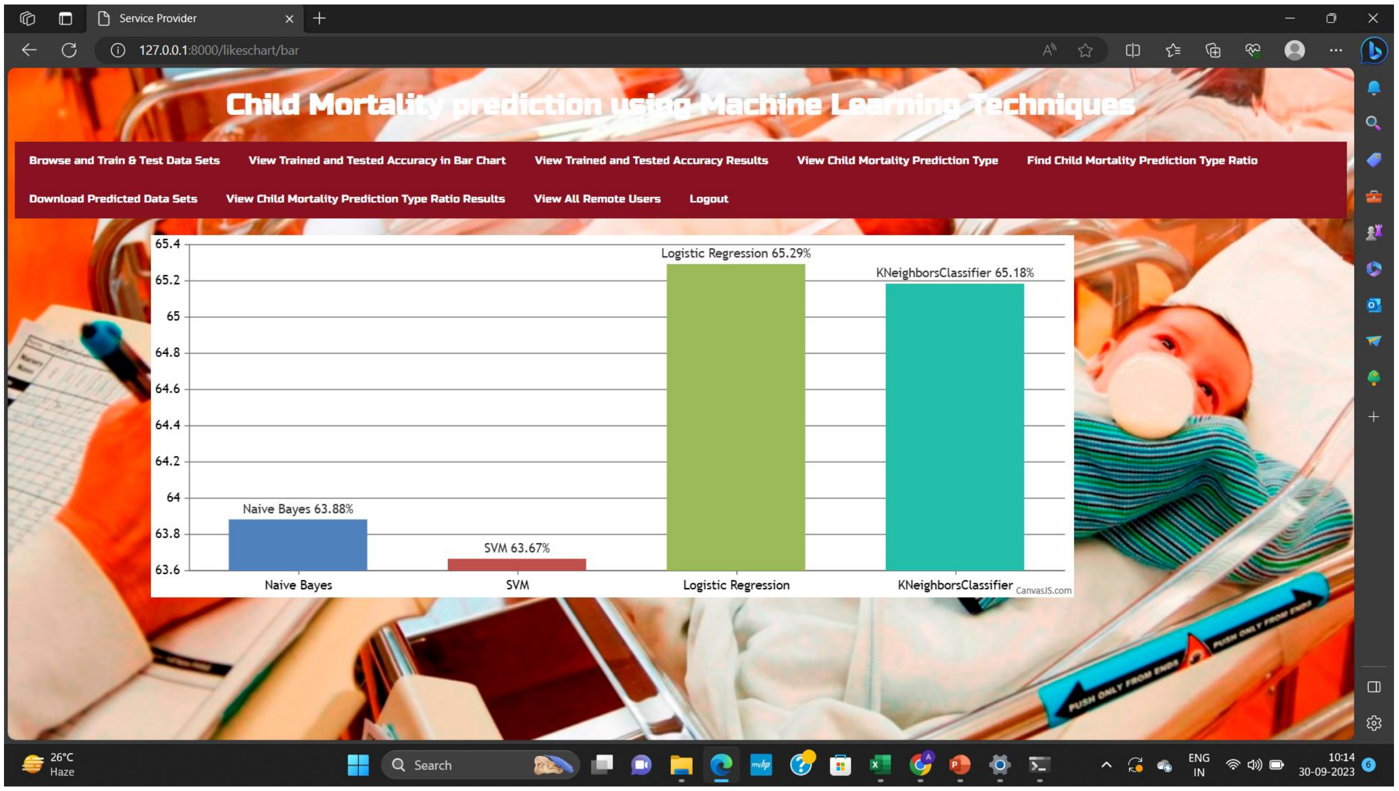
9.2 OUTPUT


Merged_document (1)_merged
Service Provider
127.0.0.1:8000/Train_Test_DataSets/
Child Mortality prediction using Machine Learning Techniques
Browse and Train & Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Child Mortality Prediction Type
Find Child Mortality Prediction Type Ratio
Download Predicted Data Sets
View Child Mortality Prediction Type Ratio Results
View All Remote Users
Logout

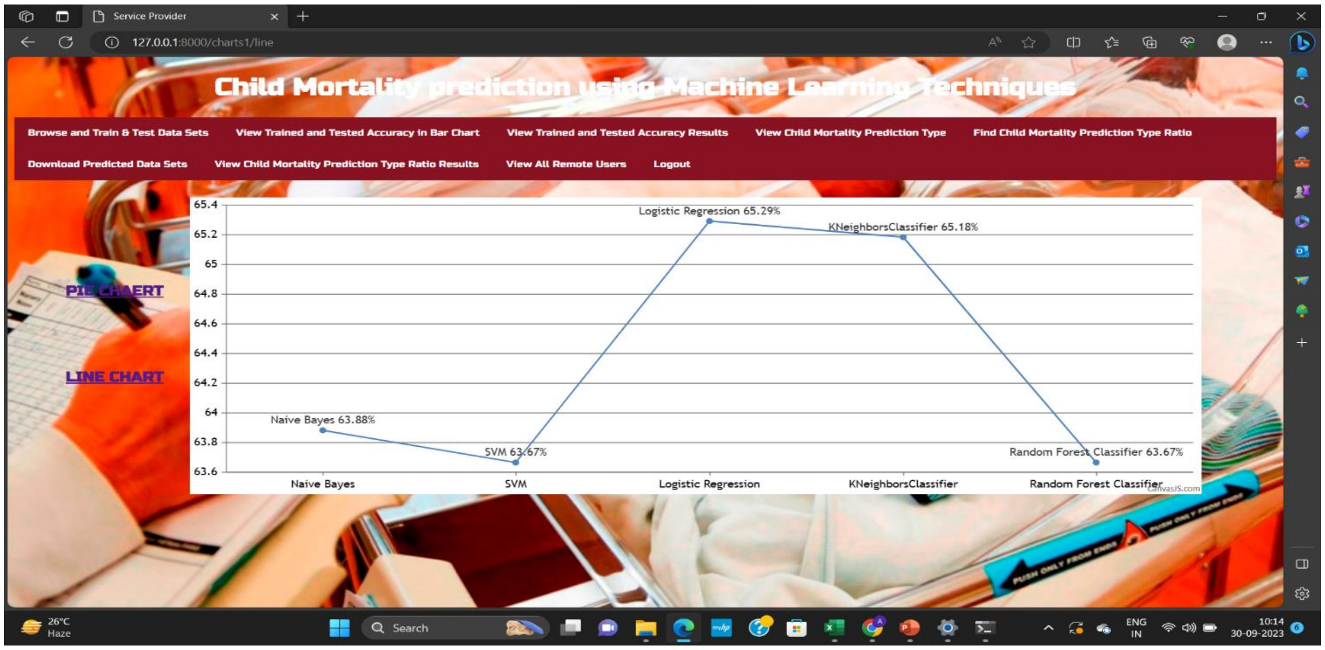
View DataSets Trained and Tested Results

Model Type	Accuracy
Naive Bayes	60.95444685466378
SVM	60.62906724511931
Logistic Regression	61.93058568329718
KNeighborsClassifier	61.82212581344903
Random Forest Classifier	60.62906724511931

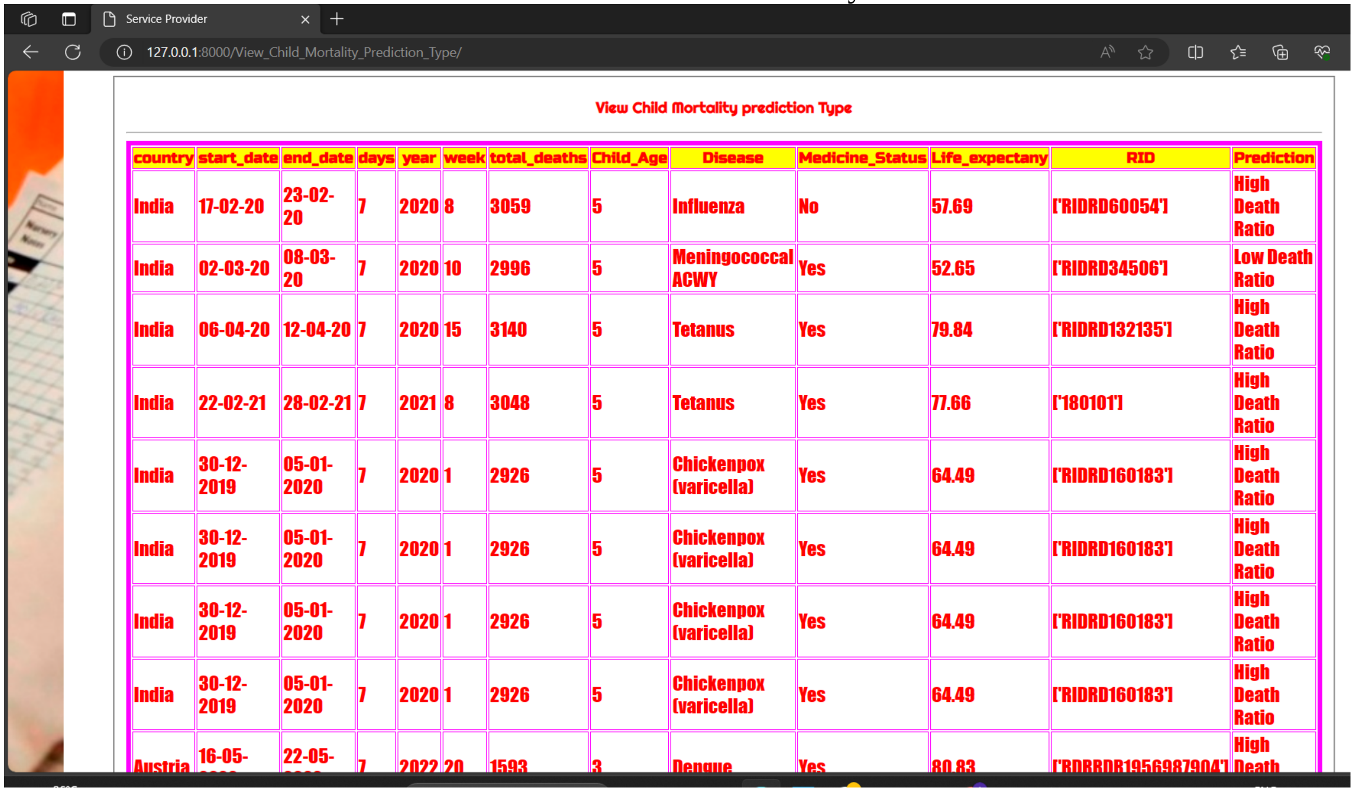
9.7 Trained and Tested Results


Service Provider
127.0.0.1:8000/likeschart/bar
Child Mortality prediction using Machine Learning Techniques
Browse and Train & Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Child Mortality Prediction Type
Find Child Mortality Prediction Type Ratio
Download Predicted Data Sets
View Child Mortality Prediction Type Ratio Results
View All Remote Users
Logout
65.4
65.2
65
64.8
64.6
64.4
64.2
64
63.8
63.6
Logistic Regression 65.29%
KNeighborsClassifier 65.18%
Naive Bayes 63.88%
SVM 63.67%
Naive Bayes
SVM
Logistic Regression
KNeighborsClassifier
CanvasJS.com
26°C Haze
Search
ENG IN
10:14 30-09-2023

9.8 Trained and Tested Accuracy in Bar Chart



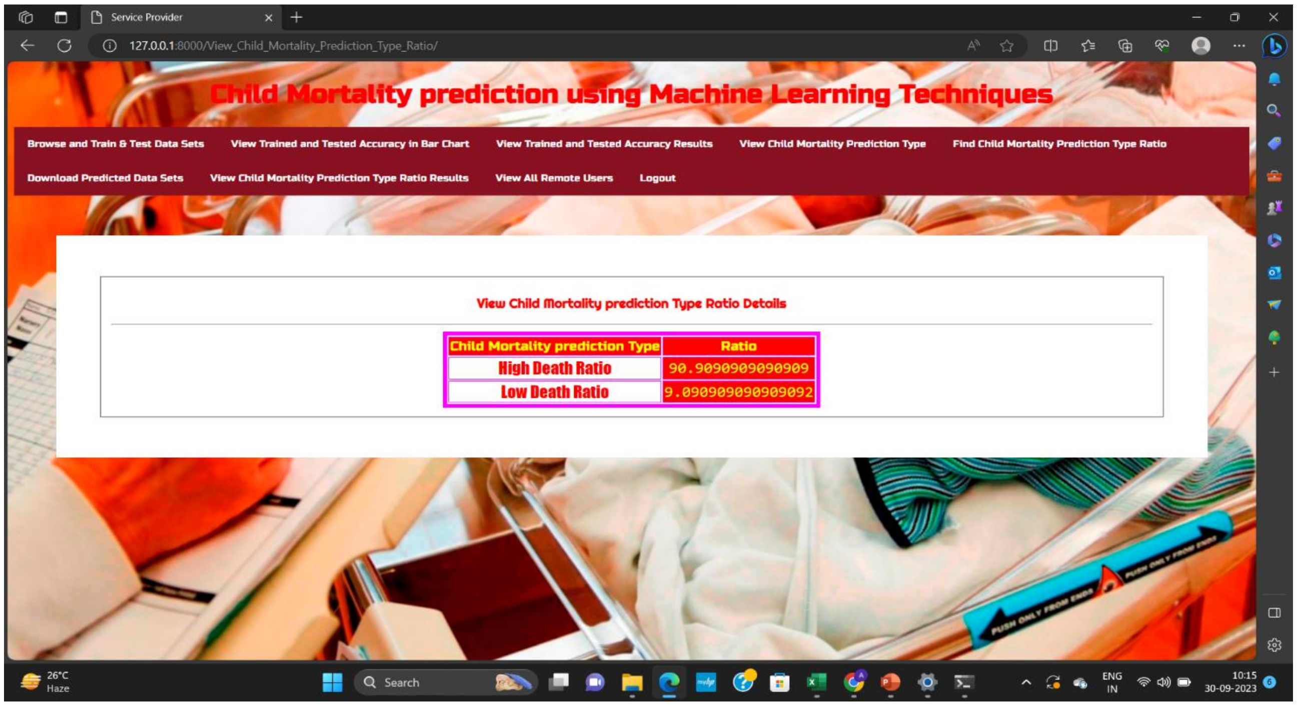
9.9 Trained and Tested Accuracy Results



View Child Mortality prediction Type

country	start_date	end_date	days	year	week	total_deaths	Child_Age	Disease	Medicine_Status	Life_expectany	RID	Prediction
India	17-02-20	23-02-20	7	2020	8	3059	5	Influenza	No	57.69	['RIDRD60054']	High Death Ratio
India	02-03-20	08-03-20	7	2020	10	2996	5	Meningococcal ACWY	Yes	52.65	['RIDRD34506']	Low Death Ratio
India	06-04-20	12-04-20	7	2020	15	3140	5	Tetanus	Yes	79.84	['RIDRD132135']	High Death Ratio
India	22-02-21	28-02-21	7	2021	8	3048	5	Tetanus	Yes	77.66	['180101']	High Death Ratio
India	30-12-2019	05-01-2020	7	2020	1	2926	5	Chickenpox (varicella)	Yes	64.49	['RIDRD160183']	High Death Ratio
India	30-12-2019	05-01-2020	7	2020	1	2926	5	Chickenpox (varicella)	Yes	64.49	['RIDRD160183']	High Death Ratio
India	30-12-2019	05-01-2020	7	2020	1	2926	5	Chickenpox (varicella)	Yes	64.49	['RIDRD160183']	High Death Ratio
India	30-12-2019	05-01-2020	7	2020	1	2926	5	Chickenpox (varicella)	Yes	64.49	['RIDRD160183']	High Death Ratio
Austria	16-05-	22-05-	7	2022	20	1593	3	Dengue	Yes	80.83	['RDRRDR1956987904']	High Death

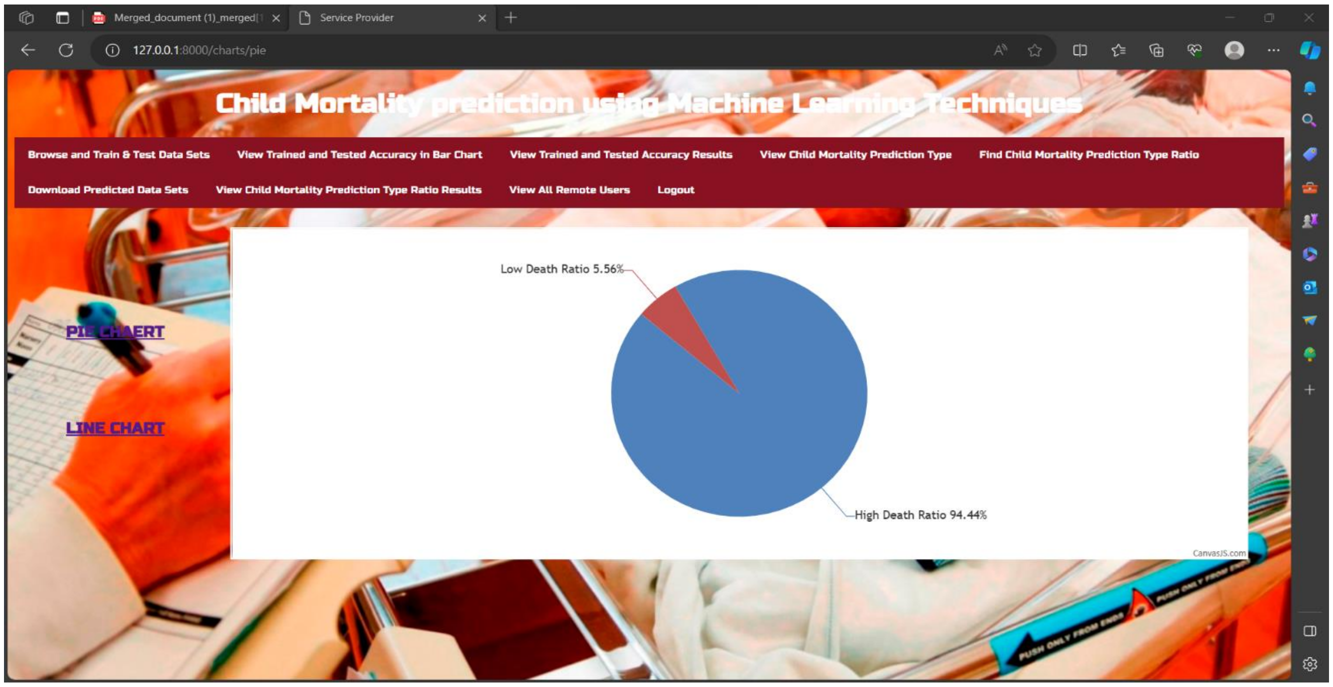
9.10 Prediction of Child Mortality Prediction Type


Service Provider
127.0.0.1:8000/View_Child_Mortality_Prediction_Type_Ratio/
Child Mortality prediction using Machine Learning Techniques
Browse and Train & Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Child Mortality Prediction Type
Find Child Mortality Prediction Type Ratio
Download Predicted Data Sets
View Child Mortality Prediction Type Ratio Results
View All Remote Users
Logout

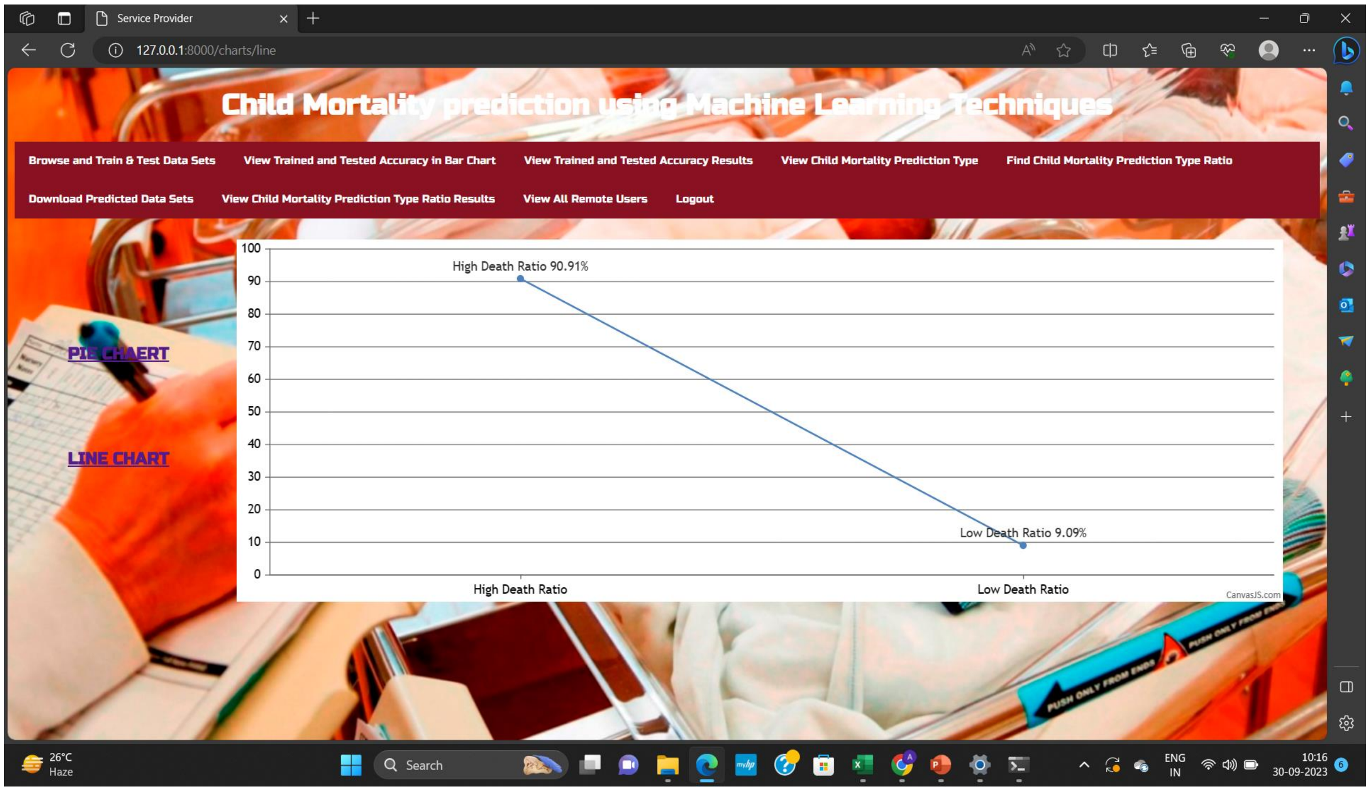
View Child Mortality prediction Type Ratio Details

Child Mortality prediction Type	Ratio
High Death Ratio	90.9090909090909
Low Death Ratio	9.090909090909092

9.11 Child Mortality Prediction Type Found Ratio


Merged_document (1)_merged
Service Provider
127.0.0.1:8000/charts/pie
Child Mortality prediction using Machine Learning Techniques
Browse and Train & Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Child Mortality Prediction Type
Find Child Mortality Prediction Type Ratio
Download Predicted Data Sets
View Child Mortality Prediction Type Ratio Results
View All Remote Users
Logout
PIE CHAERT
LINE CHART
Low Death Ratio 5.56%
High Death Ratio 94.44%
CanvasJS.com

9.12 Accuracy Results in Pie Chart


Service Provider
127.0.0.1:8000/charts/line
Child Mortality prediction using Machine Learning Techniques
Browse and Train & Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Child Mortality Prediction Type
Find Child Mortality Prediction Type Ratio
Download Predicted Data Sets
View Child Mortality Prediction Type Ratio Results
View All Remote Users
Logout
PIE CHAERT
LINE CHART
High Death Ratio 90.91%
Low Death Ratio 9.09%
High Death Ratio
Low Death Ratio
CanvasJS.com

9.13 Accuracy Results in Line-Chart

10. 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.

10.1 TYPES OF TESTS

10.1.1 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.

10.1.2 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.

Types of Integration Testing:

Top Down Integration

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

10.1.3 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.

10.1.4 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.

10.1.5 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.

10.1.6 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.

11. CONCLUSION

An ML-based approach for the spatial rainfall field estimation has been defined. By integrating heterogeneous data sources, such as RGs, radars, and satellites, this methodology permits estimation of the rainfall, where RGs are not present, also exploiting the spatial pattern recognition ensured by radars and satellites. After a phase of preprocessing, a random uniform under sampling strategy is adopted, and finally, an HPEC permits the model used to be built to estimate the severity of the rainfall events. This ensemble is based on two levels: in the first level, a set of RF classifiers are trained, while, in the second level, a probabilistic metal earner is used to combine the estimated probabilities provided by the base classifiers according to a stacking schema. Experimental results conducted on real data provided by the Department of Civil Protection show significant improvements in comparison with Kriging with external drift, a largely used and well-recognized method in the field of rainfall estimation. In particular, the ensemble method exhibits a better capacity in detecting the rainfall events. Indeed, both the POD (0.58) and the MSE (0.11) measures obtained by HPEC are significantly better than the values obtained by KED (0.48 and 0.15, respectively). As for the last two classes, representing intense rainfall events, the difference between the Kriging method and HPEC is not significant (in terms of F-measure) although HPEC is computationally more efficient.

Indeed, the complexity of the Kriging method is cubic in the number of the samples , which makes the procedure really expensive from the computational point of view, when a large number of points are analyzed. On the contrary, the ML algorithms (i.e., RF) exhibit a quadratic complexity. Moreover,
ensemble methods are highly scalable and parallelizable. Therefore, we believe that our approach has some relevant advantages in this field of application.

In addition, by analyzing the effect of the integration of the different sources of data, it is evident that all the data sources contribute to the good performance of the technique. In particular, by removing the RG information, the performance of the algorithm worsens the sensibly for all the measures. In the cases of the removal of one of the other two types of data, the degradation is less evident; however, the lowest value (0.11) of the MSE is obtained when all the data are used, which confirms that it is necessary to use all the sources of data to obtain better results.

As future work, we plan to validate the method on a larger time interval, in order to consider effects due to seasonal and yearly variability, also considering the possibility of incrementally building the flexible ensemble model with the new data. In addition, we want to evaluate the effectiveness of the algorithm in individuate highly localized heavy precipitation events, also by adopting time series analysis to analyze the individual contributions of the different features for radar and Meteosat.