PREDICTING POVERTY LEVEL USING SATELLITE IMAGERY

A PROJECT REPORT

Submitted by

SINEKA.V 211419104254

VINOTHINI.P 211419104306

SRUTHI.R 211419104270

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING
IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An autonomous institution affiliated to Anna University, Chennai)

APRIL 2023

PANIMALAR ENGINEERING COLLEGE

(An autonomous institution affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **“PREDICTING POVERTY LEVEL USING SATELLITE IMAGERY”** is the bonafide work of **“SINEKA.V [211419104254],VINOTHINI.P[211419104306],SRUTHI.R[211419104270]”** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.L.JABASHEELA , M.E., Ph.D.,	**Dr.K.VALARMATHI M.E,Ph D.,**
HEAD OF THE DEPARTMENT	**PROFESSOR SUPERVISOR**
DEPARTMENT OF CSE,	DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,	PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,	NASARATHPETTAI,
POONAMALLEE,	POONAMALLEE,
CHENNAI-600 123.	CHENNAI-600 123.

Certified that the above candidates were examined in the Anna University Project Viva-Voce Examination held on...........................

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION BY THE STUDENT

We, SINEKA.V (211419104254), VINOTHINI.P (211419104306), SRUTHI.R (211419104270) hereby declare that this project report titled **“PREDICTING POVERTY LEVEL USING SATELLITE IMAGERY”,** under the guidance of Dr.K.VALARMATHI M.E,Ph D., is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

SINEKA.V
VINOTHINI.P
SRUTHI.R

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our beloved Directors **Tmt.C.VIJAYARAJESWARI**, **Dr.C.SAKTHI KUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. L.JABASHEELA M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my **Project Guide Dr.K.VALARMATHI M.E,Ph D.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

SINEKA.V

VINOTHINI.P

SRUTHI.R

ABSTRACT

Determining the poverty levels of various regions throughout the world is crucialin identifying interventions for poverty reduction initiatives and directing resources fairly. However, reliable data on global economic livelihoods is hard to come by, especially for areas in the developing world, hampering efforts to both deploy services and monitor/evaluate progress. This project is to create a model to predict the poverty levels of an area by using a satellite images can be achieved through remote sensing methods. Specifically, satellite images processed through convolutional neural networks have shown promise in predicting the intensity of night time lights, which can then be used to gauge the underlying poverty level .This will help philanthropic agencies and government to identify where resources and interventions are needed and help guide the direction of financial aid and Frequent and reliable data on poverty levels and distribution allows agencies to better track progress on the Sustainable Development Goals. This project proposes to use satellite images to detect economic activity and, as a result, estimate poverty in a location. A Recurrent neural network is trained to learn various developmental parameters like rooftop type, source of lighting and proximity to water sources, Agriculture Areas, Road Structure, and Industrial Areas.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

People in a country should know about the economic status of their own. For thesocial Analyst, it is very important to know about the poverty data to calculate theeconomic condition of a particular region. From this project, we just calculate thepoverty level of a region using satellite images. This poverty level data will be useful for the government also to take appropriate steps to upgrade the region whether the level of living i.e., poverty level is low. For calculating the poverty level we used the datasets with multiple data and satellite images.By using this, the machine trained and will find out the value for anypart of the country in the world using a deep learning Algorithm “Recurrent Neural Network” with the high Accuracy of 9.8 .

1.1 OVERVIEW

The project comprises of detecting the poverty level by using a atellite imagery Using a recurrent neural network.The dataset is in image format that is satellite image of a region. Several libraries also used such as matplotlib etc. divided the initial images into training, tuning, and testing sets.The splits were executed on the country level to ensure that data from the same country is not in more than one set. Furthermore, the splits were consistent across daytime and night time images meaning that daytime and night time images of the same region were in the same split, ensuring consistency in reporting metrics across daytime and night time experiments All evaluation is done on held-out test regions that were not used in model development. This mimics the real-world usage of such a network, where the network is expected to make predictions on regions unseen during training. After that clustering will happen based on the number we gave and then predicting process will be done it will tell the wealth of the region.

1.2 PROBLEM STATEMENT

This project is to create a model to predict the poverty levels of an area by using a satellite images can be achieved through remote sensing methods. Specifically, satellite images processed through convolutional neural networks have shown promise in predicting the intensity of night time lights, which can then be used to gauge the underlying poverty level .This will help philanthropic agencies and government to identify where resources and interventions are needed and help guide the direction of financial aid and Frequent and reliable data on poverty levels and distribution allows agencies to better track progress on the Sustainable Development Goals.

CHAPTER 2

LITERATURE SURVEY

TITLE: Satellite-Based Mapping of Urban Poverty with Transfer-Learned Slum Morphologies

AUTHOR: Thomas Stark, Michael Wurm, Xiao Xiang Zhu and Hannes Taubenbock.

YEAR:2023

DESCRIPTION: Satellite-based mapping can provide valuable information about slums where insights about the location and size are still missing. Large-scale slum mapping remains a challenge, fuzzy feature spaces between formal and informal settlements, significant imbalance of slum occurrences opposed to formal settlements, and various categories of multiple morphological slum features. We propose a transfer learned fully convolutional Xception network (XFCN), which is able to differentiate between formal built-up structures and the various categories of slums in high-resolution satellite data.

DISADVANTAGES: By using a large-scale globally distributed dataset of slums, the FCN is better able to generalize and, thus, is able to map slums in areas where this was previously not possible on high-resolution remote sensing data.

TITLE : Using Convolutional Neural Networks on Satellite Images to Predict Poverty

AUTHOR: Arwa Okaidat, Shatha Melhem, Heba Alenezi and Rehab M. Duwairi

YEAR:2022

DESCRIPTION: This paper focuses on Africa as it is considered the poorest continent. The data, we have used, consist of three datasets which contain satellite images for three countries in Africa with different levels of poverty: Ethiopia, Malawi, and Nigeria. In order to classify the satellite images, two pre-trained Convolutional Neural Networks models (ResNet50 and VGG16) were implemented in addition to our novel structure of CNN.

DISADVANTAGES: The process of going around rural areas and manually tracking census data is time-consuming, needs a lot of human effort, and is expensive.

TITLE: Poverty Level Prediction Based on E-Commerce Data Using K-Nearest Neighbor and Information-Theoretical-Based Feature Selection.

AUTHOR: Tiara Fatehana Aulia, Dedy Rahman Wijaya, Elis Hernawati,Wahyu Hidayat

YEAR: 2022

DESCRIPTION: In this very rapid development, many methods can be used to determine the poverty level. One of them is with the use of the rapid development of E-commerce in Indonesia, which can determine the level of poverty in Indonesia. In this study, we proposed a method to predict the poverty level based on an e-commerce dataset using K-Nearest Neighbor and Information Theoretical Based Feature Selection.

DISADVANTAGES: The algorithm relies on majority voting based on class membership of k-nearest samples, so the normalization of data is required to make correct predictions.

TITLE: Predicting Poverty through Machine Learning and Satellite Images

AUTHOR: Yan Xiao

YEAR:2021

DESCRIPTION: In this research, I develop a machine learning model that leverages transfer learning, deep learning, and random forest algorithm to predict the poverty level of three African countries based on satellite images. I extracted features from satellite images through VGG-11 network and then feed them into random forest model. Furthermore, I implemented a perturbation based algorithm occlusion in Septum package to explore the feature importance of CNN model and used locally linear embedding to visualization the distribution of extracted features from different regions.

DISADVANTAGES: The process of going around rural areas and manually tracking census data is time consuming.

TITLE: Poverty Prediction Through Machine Learning

AUTHOR: Huang Zixi

YEAR:2021

DESCRIPTION: The paper considers poverty as an outcome of multidimensional factors, and offers various practical models for such prediction using machine learning, none of which accounts for the whole, while some factors may outweigh others. Thereby, an integrated approach of prediction is needed by combining the data from Poverty Probability Index and Oxford Poverty & Human Development Initiative. Through applying linear regression model, decision tree, random forest model, gradian boosting model, and neural network to analysis existing data, the paper assesses respectively the extent to which the factors matter and the efficacy of each model.

DISADVANTAGES: The process of going around rural areas and manually tracking census data is time consuming.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Currently, poverty is formally calculated by numerous philanthropic agencies including the World Bank. One of the reasons why data on poverty is sparse in the developing world is because it is infrequently collected due to the high cost associated with on-the-ground surveys. In existing with the E-Commerce Data Using K-Nearest Neighbor algorithm they can find the poverty level on county that is Indonesia. It cannot find any other country and accuracy level is also low.

DISADVANTAGES:

- It is a time-consuming process.
- shows lower accuracy rate.

3.2 PROPOSED SYSTEM

More specifically, both daytime and nighttime satellite imagery of regions canbe used to estimate poverty in certain regions. Deep learning has been a main factor behind recent breakthroughs in numerous computer vision tasks such as image classification, segmentation, and object detection. In this project, we assemble a dataset of 88,386 images from 44,193 cities spanning Africa, South America, Asia, Europe, and the Caribbean satellite image. For each city, we obtain a daytime satellite image, a nighttime satellite image, and the city's wealth index. Then train Recurrent neural networks (RNNs) to predict a city'swealth index, given a satellite image.

ADVANTAGES:

- It is time efficient process.
- more accuracy rates

3.3 FEASIBILITY STUDY

The objective of feasibility study is not only to solve the problem but also to acquire a sense of its scope. During the study, the problem definition was crystallized and aspects of the problem to be included in the system are determined. Consequently, benefits are estimated with greater accuracy at this stage. The key considerations are:

- Economic feasibility
- Technical feasibility
- Operational feasibility

Economic Feasibility

Economic feasibility studies not only the cost of hardware, software is included but als the benefits in the form of reduced costs are considered here. This project, if installed will certainly be beneficial since there will be reduction in manual work andincrease in the speed of work.

Total number of lines of code (LOC)=204

KSLOC=204/1000=0.204

Effort=2.4*(0.204) ^1.05=0.452 person-month

Development time=2.5*(0.452192044) ^0.38=1.849 month

Average staff size=0.452192044/ 1.84910987162=0.24454 person

Productivity=0.204/0.452192044=0.451 KSLOC/person-month

P=451 LOC/person-month

Technical Feasibility

Technical feasibility evaluates the hardware requirements, software technology, available personnel etc., as per the requirements it provides sufficient memory to hold and process.

1. Deep learning Algorithm – Recurrent neural network
2. Deep learning
3. Anaconda navigator
4. Anaconda prompt

Social Feasibility

Software requirement specification is a kind of document which is created by a software analyst after the requirements collected from the various sources - the requirement received by the customer written in ordinary language. It is the job of the analyst to write the requirement in technical language so that they can be understood and beneficial by the development team.

1. Providing the poverty level of the particular place with high Accuracy.
2. people and analyst can get beneficiaries

SYSTEM REQUIREMENTS

3.4 HARDWARE REQUIREMENTS:

System	: Pentium IV 2.4 GHz.
Hard Disk	: 40 GB.
Floppy Drive	: 1.44 Mb.
Monitor	: 15 VGA Colour.
Mouse	: Logitech.
Ram	: 512 Mb.

3.5 SOFTWARE REQUIREMENTS:

Operating system	:	Windows.
Coding Language	:	Python 3.8
Database	:	MYSQL

CHAPTER 4

SYSTEM DESIGN

Step-by-step progression through a procedure or system especially using connecting lines and a set of conventional symbols

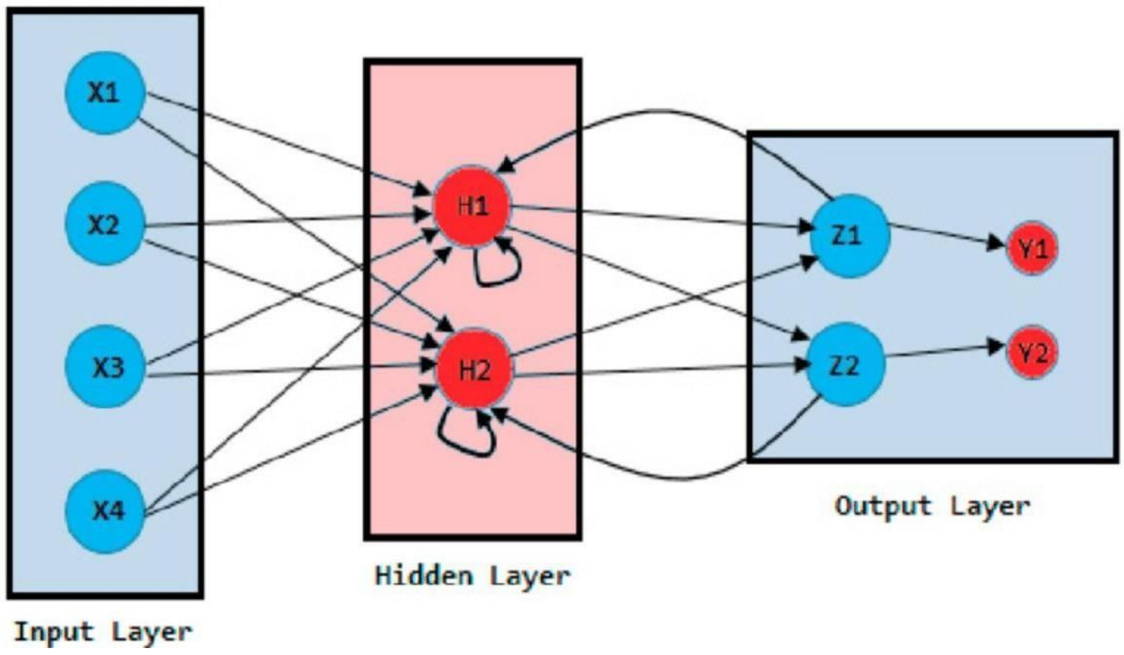


Fig 4.1 System Design

PYTHON

4.1 OVERVIEW

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a general-purpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific problems. Python programs are platform independent because they can be run on different platforms using an interpreter built specifically for that platform.

4.2 WORKING OF PYTHON

Python is an object-oriented programming language like Java. Python is called an interpreted language. Python uses code modules that are interchangeable insteadof a single long list of instructions that was standard for functional programming languages. The standard implementation of python is called “cpython”.

Python is an object-oriented programming language like Java. Python is called 13 an interpreted language. Python uses code modules that are interchangeable instead of a single long list of instructions that was standard for functional programming languages. The standard implementation of python is called “cpython”. It is the default and widely used implementation of Python.

Python doesn’t convert its code into machine code, something that hardware can understand. It actually converts it into something called byte code. So within python, compilation happens, but it’s just not into a machine language. It is into byte code (.pycor .pyo) and this byte code can’t be understood by the CPU. So we need an interpreter called the python virtual machine to execute the byte codes.

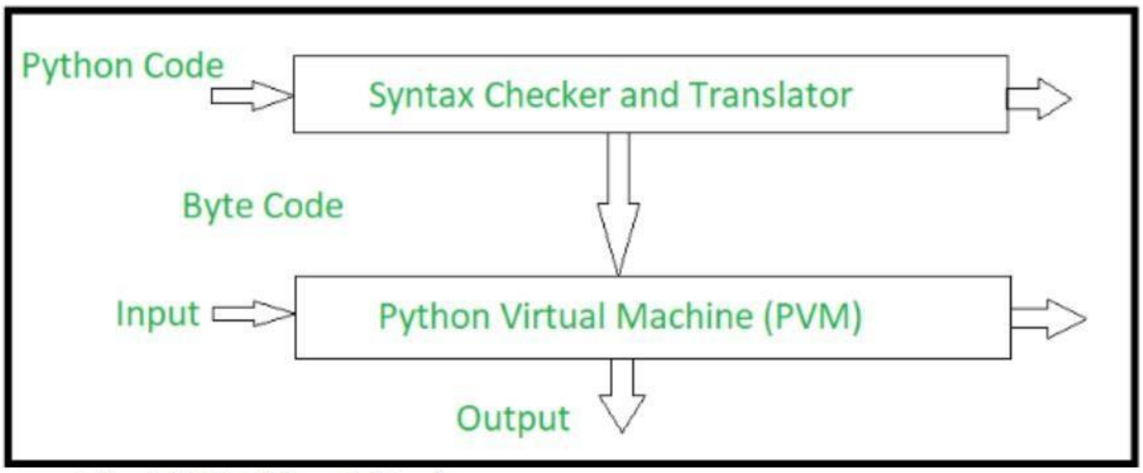


Fig 4.2 Working Of Python

The Python source code goes through the following to generate an executable code:

- Step 1: The python compiler reads a python source code or instruction. Then it verifies that the instruction is well-formatted, i.e., it checks the syntax of each line. If it encounters an error, it immediately halts the translation and shows an error message.

- Step 2: If there is no error, i.e. if the python instruction or source code is well formatted then the compiler translates it into its equivalent form in an intermediate language called “Byte code”.

- Step 3: Byte code is then sent to the Python Virtual Machine (PVM) which is the python interpreter. PVM converts the python byte code into machine-executable code. If an error occurs during this interpretation, then the conversion is halted with an error message.

CHAPTER 5

SYSTEM DESIGN DIAGRAM

5.1 FLOW CHART DIAGRAM

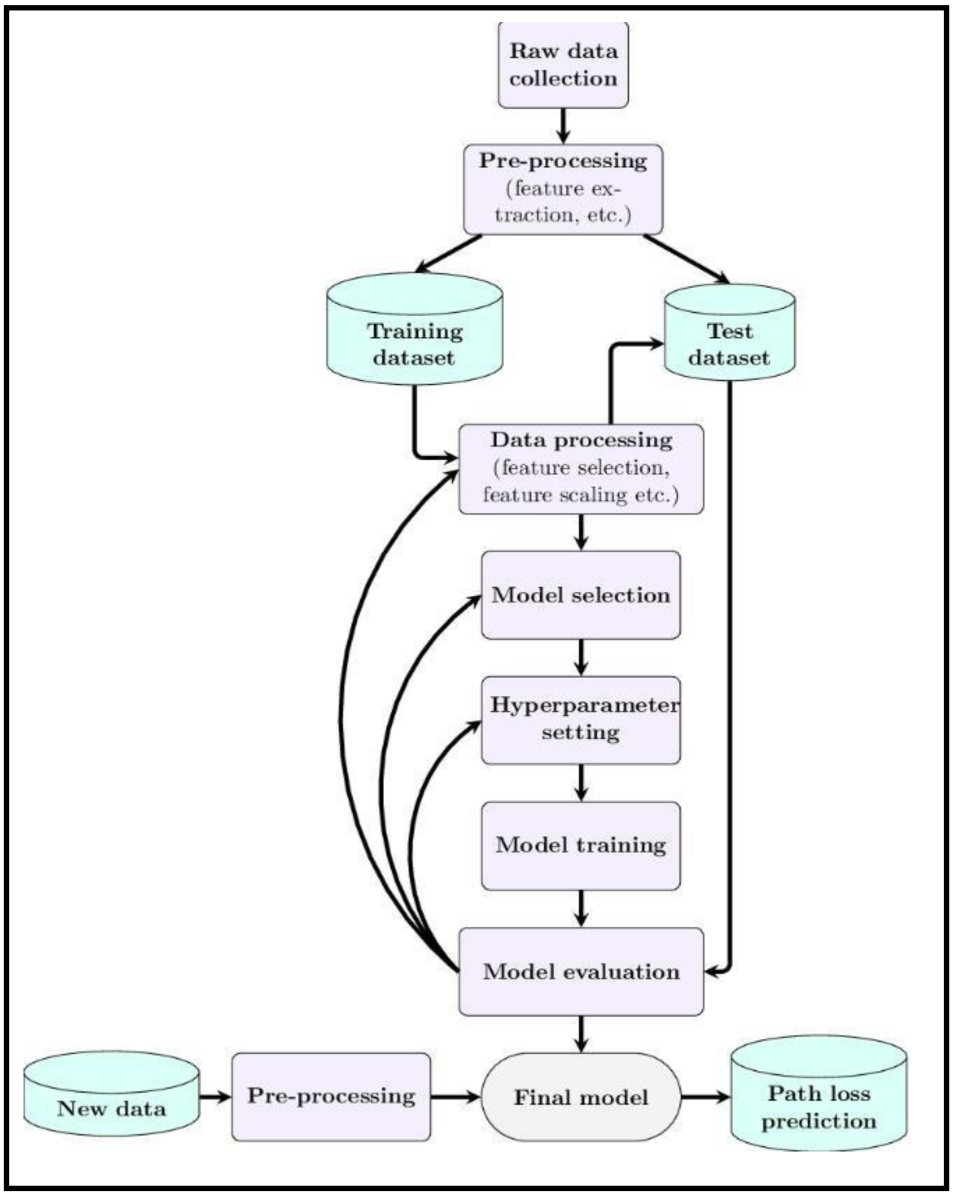


Fig 5.1 Flow Chart Diagram

5.2 USECASE DIAGRAM

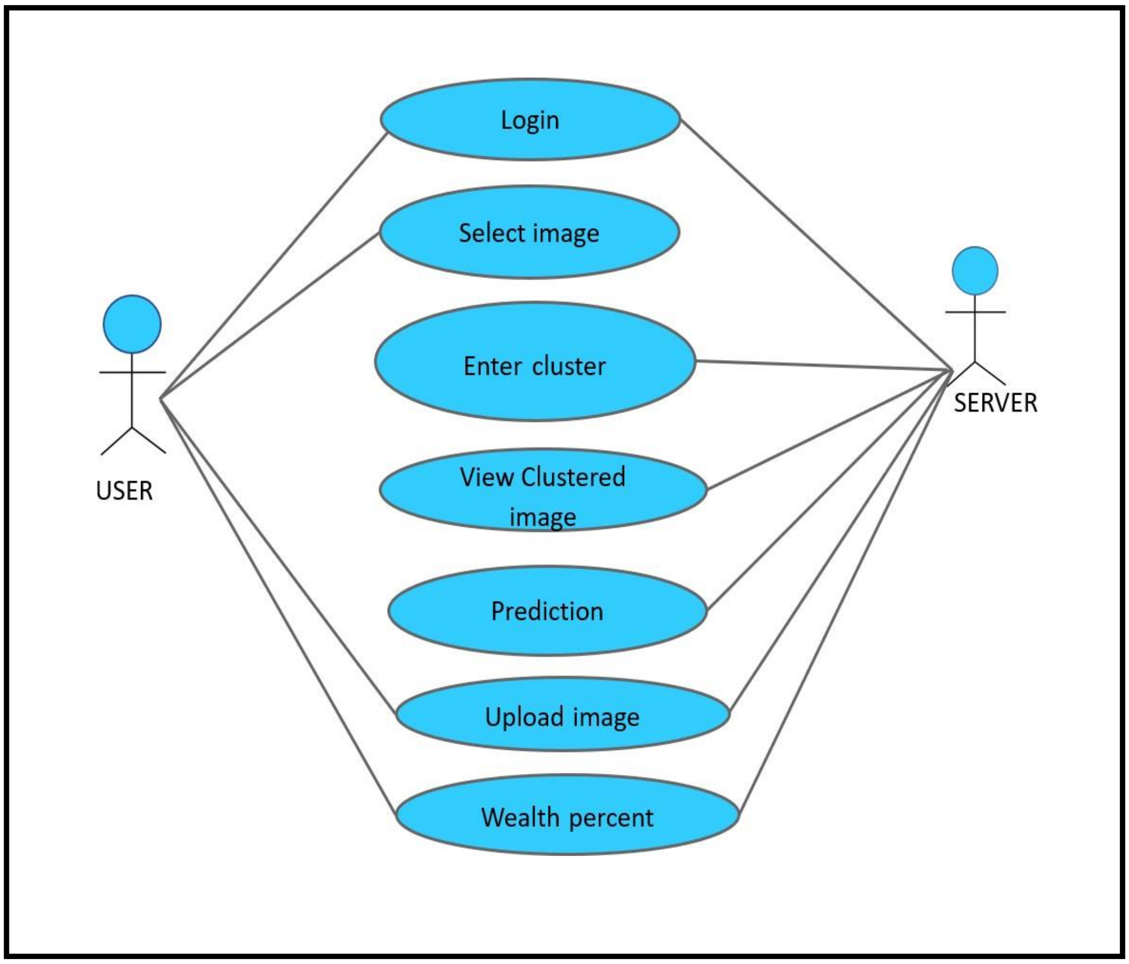


Fig 5.2 Use case Diagram

5.3 ACTIVITY DIAGRAM

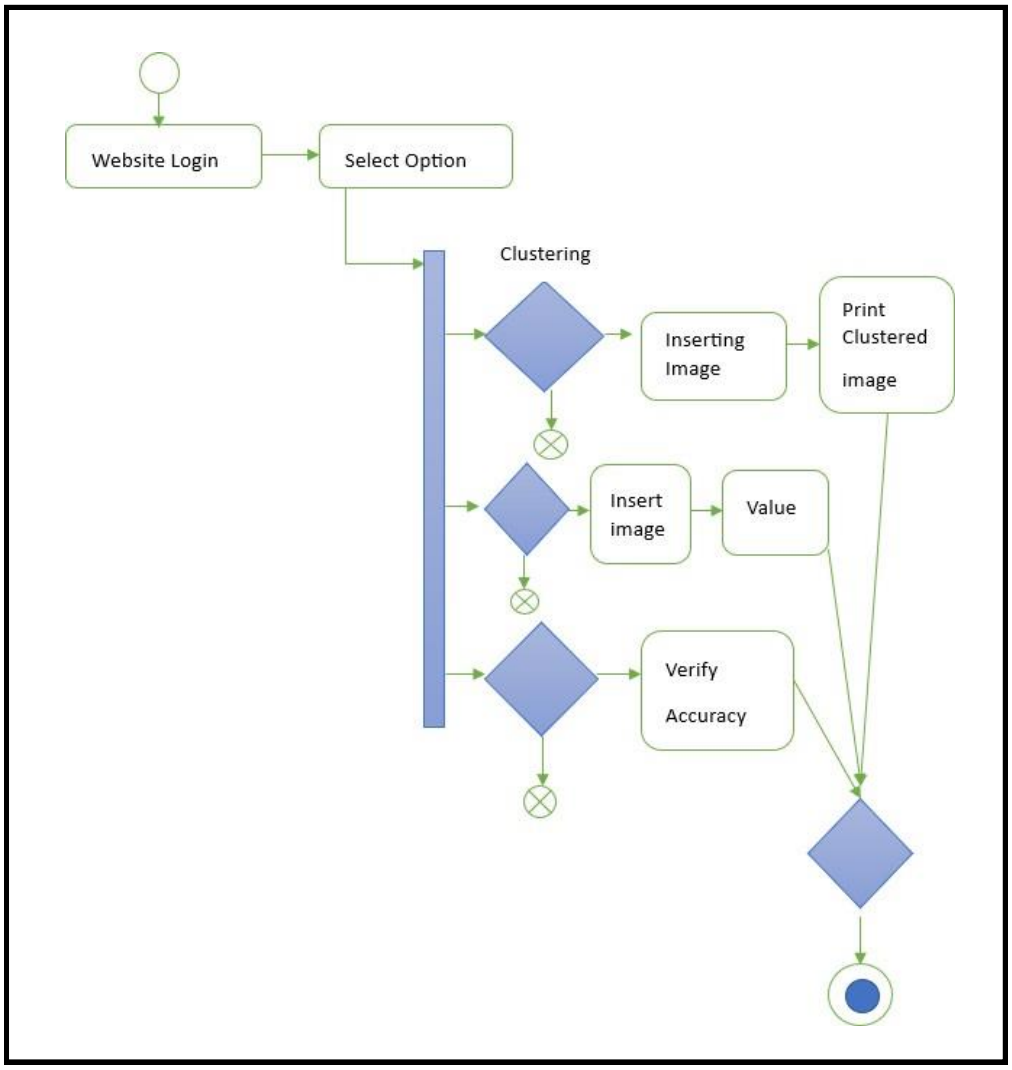


Fig 5.3 Activity Diagram

5.4 SEQUENCE DIAGRAM

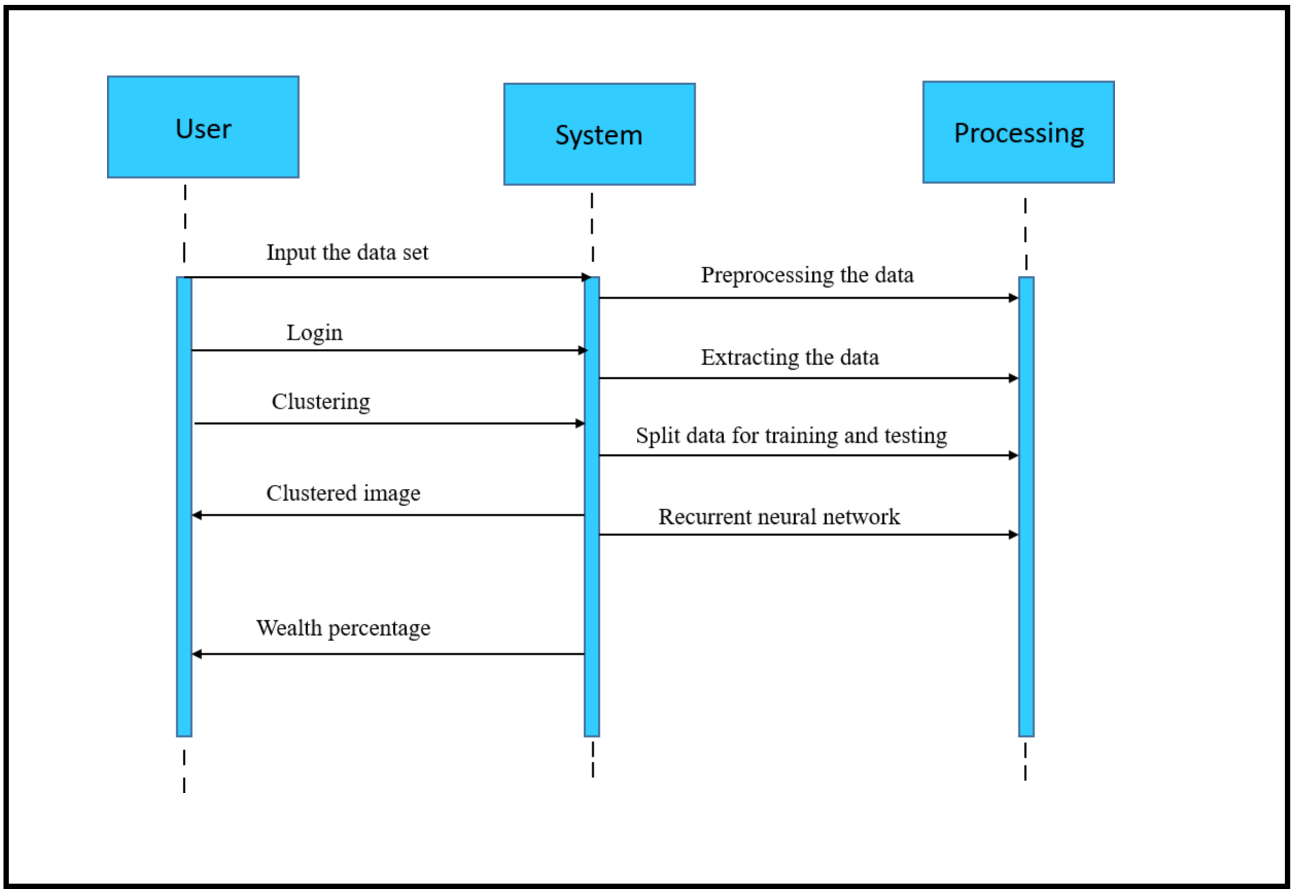


Fig 5.4 Sequence Diagram

CHAPTER 6

SYSTEM ARCHITECTURE

6.1 ARCHITECTURE OVERVIEW

Machine Learning is, undoubtedly, one of the most exciting subsets of Artificial Intelligence. It completes the task of learning from data with specific inputs to the machine. It’s important to understand what makes Machine Learning work and, thus, how it can be used in the future.

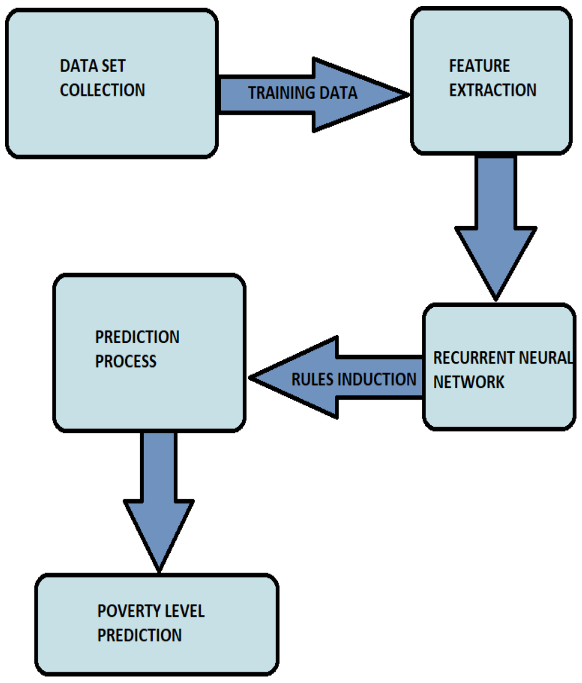


Fig 6.1 System Architecture

The Machine Learning process starts with inputting training data into the selected algorithm. Training data being known or unknown data to develop the final Machine Learning algorithm. The type of training data input does impact the algorithm, and that concept will be covered further momentarily. New input data is fed into the machine learning algorithm to test whether the algorithm works correctly. The prediction and results are then checked against each other.If the prediction and results don’t match, the algorithm is re-trained multiple times until the data scientist gets the desired outcome. This enables the machine learning algorithm to continually learn on its own and produce the optimal answer, gradually increasing in accuracy over time.

6.2 RECURRENT NEURAL NETWORK

Recurrent Neural Network (RNN) is a type of Neural Network where the **output from the previous step are fed as input to the current step**. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus, RNN came into existence, which solved this issue with the help of a Hidden Layer. The main and most important feature of RNN is **Hidden state**, which remembers some information about a sequence.

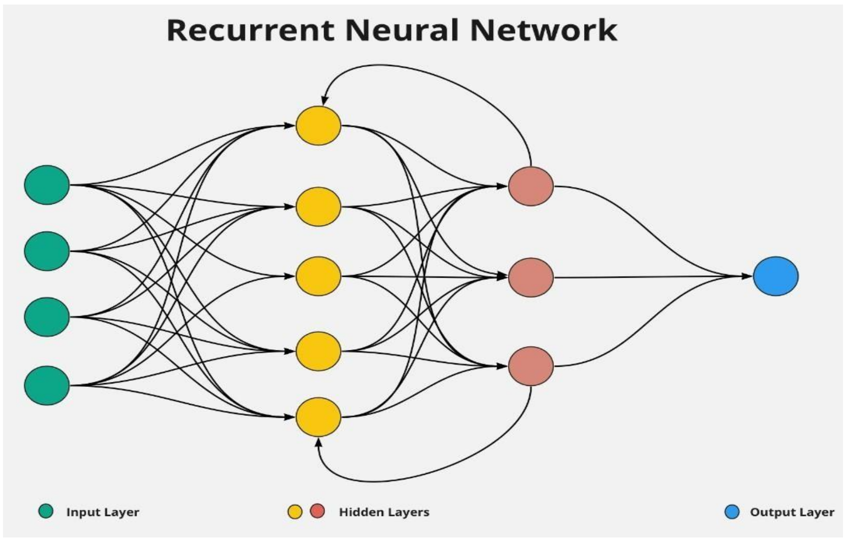


Fig 6.2.1 RNN

RNN have a **“memory”** which remembers all information about what has been calculated. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

Training through RNN

- A single-time step of the input is provided to the network.
- Then calculate its current state using a set of current input and the previous state.
- The current ht becomes ht-1 for the next time step.
- One can go as many time steps according to the problem and join the information from all the previous states.
- Once all the time steps are completed the final current state is used to calculate the output.
- The output is then compared to the actual output i.e the target output and the error is generated.
- The error is then back-propagated to the network to update the weights and hence the network (RNN) is trained.

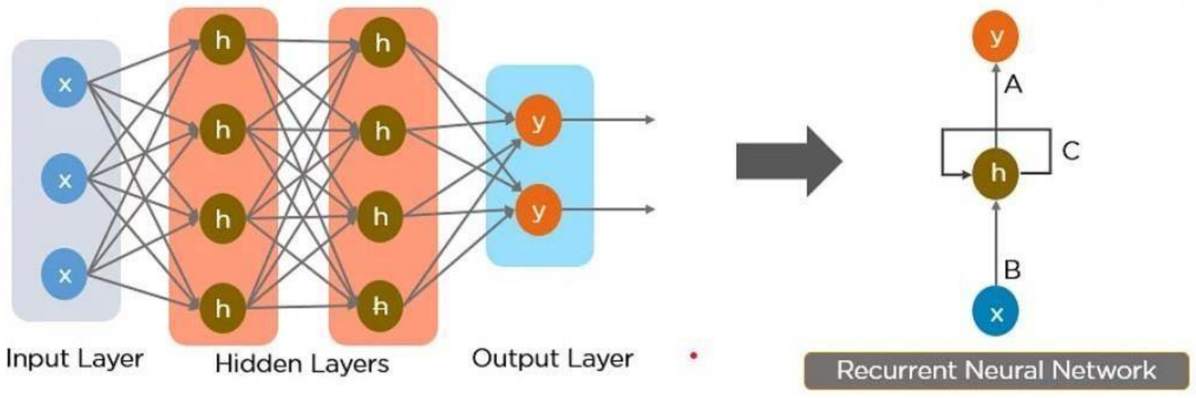


Fig 6.2.2 RNN Architecture

Advantages of Recurrent Neural Network

1. An RNN remembers each and every piece of information through time. It is useful in time series prediction only because of the feature to remember previous inputs as well. This is called Long Short Term Memory.
2. Recurrent neural networks are even used with convolutional layers to extend the effective pixel neighbourhood.

Disadvantages of Recurrent Neural Network

1. Gradient vanishing and exploding problems.
2. Training an RNN is a very difficult task.
3. It cannot process very long sequences if using tanh or rely as an activation function.

DATASET

A dataset in machine learning is, quite simply, a collection of data pieces that canbe treated by a computer as a single unit for analytic and prediction purposes. This means thatthe data collected should be made uniform and understandable for a machine that does not see data the same way as humans do.

6.3 MODULES

Home Page

After running the project code in anaconda prompt ,a URL link will be displayed in the terminal .We have to copy that and paste it in any browsing website. At that time the project website will be opened. The home page contains the full details of the project and having the login button by which we can get into the project with appropriate password.

Login Page

The login page contains the username and password input boxes in which we have to give the appropriate data to open into the next page.

Clustering Page

In this page ,we have to give a image input to cluster the places based on some criteria and map them with different colours. This page will give a good visualization for the people for better understanding about the region.

Prediction Page

This will be the main core page, In which we have to give a image as input and the machine will predict the wealth of that area using recurrent neural network. And give a wealth value at the bottom of the page.

Analysis Page

In this page, the project analysis i.e., the accuracy map and some analysis taken part .

CHAPTER 7

SYSTEM IMPLEMENTATION

app.py

```
from keras.models import model_from_json
from tensorflow.keras.utils import img_to_array
from keras.applications import imagenet_utils
from PIL import Image
import numpy as np
import flask
from flask import render_template, redirect
from flask import request, url_for, render_template, redirect
import io
import tensorflow as tf
import os
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'

from flask import render_template
from flask import request
import image_fuzzy_clustering as fem
import os
import secrets
from PIL import Image
from flask import url_for, current_app

# initialize our Flask application and the Keras model
app = flask.Flask(__name__)
```

```
model = None

UPLOAD_FOLDER = os.path.join(app.root_path ,'static','img')
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

def load_model():
    # load the pre-trained Keras model (here we are using a model
    # pre-trained on ImageNet and provided by Keras, but you can
    # substitute in your own networks just as easily)
    global model
    #model = ResNet50(weights="imagenet")
    #model = tf.keras.models.load_model('model.h5')
    with open('model/model.json', 'r') as f:
        model = model_from_json(f.read())
    model.load_weights('model/weights.hdf5')

load_model()
global graph
graph = tf.compat.v1.get_default_graph()

def prepare_image(image, target):
    # if the image mode is not RGB, convert it
    if image.mode != "RGB":
        image = image.convert("RGB")

    # resize the input image and preprocess it
    image = image.resize(target)
    image = img_to_array(image)
    image = np.expand_dims(image, axis=0)
    image = imagenet_utils.preprocess_input(image)

    # return the processed image
    return image




@app.route('/')
@app.route('/first')
def first():
    return render_template('first.html')
@app.route('/login')
def login():
    return render_template('login.html')

@app.route('/upload')
def upload():
    return render_template('index1.html')

@app.route('/success', methods=['POST'])
def success():
    if request.method == 'POST':
        i=request.form.get('cluster')
        f = request.files['file']
        fname, f_ext = os.path.splitext(f.filename)
        original_pic_path=save_img(f, f.filename)
        destname = 'em_img.jpg'
        fem.plot_cluster_img(original_pic_path,i)
    return render_template('success.html')

def save_img(img, filename):
    picture_path = os.path.join(current_app.root_path, 'static/images', filename)
    # output_size = (300, 300)
    i = Image.open(img)
    # i.thumbnail(output_size)
    i.save(picture_path)

    return picture_path



@app.route('/chart')
def chart():
    return render_template('chart.html')

@app.route('/wealth')
def wealth():
    return render_template('wealth.html')



@app.route("/index", methods=["POST","GET"])
```

```
def predict():
    # initialize the data dictionary that will be returned from the view
    #data = {"success": False}
    data = {"success": "Upload — Satellite Image"}
    title = "Predicting Poverty"
    name = "default.png"
    # ensure an image was properly uploaded to our endpoint
    if flask.request.method == "POST":
        if flask.request.files.get("image"):

            image1 = flask.request.files["image"]
            # save the image to the upload folder, for display on the webpage.
            image = image1.save(os.path.join(app.config['UPLOAD_FOLDER'],
image1.filename))

            # read the image in PIL format
            with open(os.path.join(app.config['UPLOAD_FOLDER'],
image1.filename), 'rb') as f:
                image = Image.open(io.BytesIO(f.read()))

            # preprocess the image and prepare it for classification
            processed_image = prepare_image(image, target=(200, 200))

            '''
            # classify the input image and then initialize the list
            # of predictions to return to the client
            with graph.as_default():
                preds = model.predict(processed_image)
                results = imagenet_utils.decode_predictions(preds)
                data["predictions"] = []

            # loop over the results and add them to the list of
            # returned predictions
            for (imagenetID, label, prob) in results[0]:
                r = {"label": label, "probability": float(prob)}
                data["predictions"].append(r)
            '''

            def get_luminosity(category):
                luminosity = None
                if category == 0: # dim
                    luminosity = np.random.randint(0, 3)
                elif category == 1: # medium
                    luminosity = np.random.randint(3, 35)
                elif category == 2: # bright
                    luminosity = np.random.randint(35, 64)
                return luminosity

            def predict_wealth(luminosity):
                slope = 0.07268134920271797
                intercept = -0.3465173128978627
                return np.around(luminosity * slope + intercept, decimals=2)

            index_label_dict = {0: "DIM", 1: "MEDIUM", 2: "BRIGHT"}
            # load_model()
```

```
        #global graph
        #graph = tf.compat.v1.get_default_graph()

        with graph.as_default():
            json_file = open('model/model.json','r')
            load_model_json = json_file.read()
            json_file.close()
            load_model = model_from_json(load_model_json)
            #load weights into new model
            load_model.load_weights("model/weights.hdf5")
            print("Loaded Model from disk")
            #compile and evaluate loaded model

load_model.compile(loss='sparse_categorical_crossentropy',optimizer='adam',m
etrics=['accuracy'])
            # perform the prediction
            preds = load_model.predict(processed_image)
            print(preds)
            #print(class_names[np.argmax(preds)])
            # convert the response to a string
            #response = class_names[np.argmax(preds)]
            #return str(response)
            category = np.argmax(preds)
            label = "Predicted Wealth Classification:
{}".format(index_label_dict[category])
            label_wealth = "Predicted wealth index:
{}".format(str(predict_wealth(get_luminosity(category))))
            prob = np.max(preds)
            r = {"label": label, "probability": prob, "label_wealth": label_wealth}


            data["predictions"] = []
            data["predictions"].append(r)


        # indicate that the request was a success
        data["success"] = "Wealth Predictions"
        title = "predict"


        return render_template('index.html', data=data, title = title, 
name=image1.filename)
    # return the data dictionary as a JSON response
    return render_template('index.html', data = data, title=title, name=name)
# if this is the main thread of execution first load the model and
# then start the server


if__name__== "__main__":
    print(("* Loading Keras model and Flask starting server..."
        "please wait until server has fully started.(60sec)"))
    #load_model()
    #global graph
    #graph = tf.get_default_graph(), outdated
    #graph = tf.compat.v1.get_default_graph()
    app.run(debug=True)
```

CHAPTER 8

SYSTEM TESTING

The software, which has been developed, must be tested to prove its validity. Testing is the least creative phase of the whole cycle of system design. In the real sense it is the phase, which helps to bring out the creativity of the other phases makes it shine.

Unit Testing

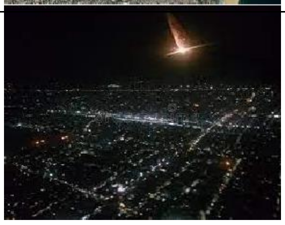
This is the first level of testing. In these different modules are tested against the specifications produced during the design of the module. During this testing the number of arguments is compared to input parameters, matching of parameter and arguments etc. It is also ensured whether the file attributes are correct, whether the Files are opened before using, whether Input/output errors are handled etc. Unit Test is conducted using a Test Driver usually.

Integration Testing

Integration testing is a systematic testing for constructing the program structure, while at the same time conducting test to uncover errors associated within the interface. Bottom-up integration is used for this phase. It begins construction and testing with atomic modules. This strategy is implemented with the following steps.

- Low-level modules are combined to form clusters that perform a specific software sub function.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure.

Table 8.1 System Testing

TEST CASE ID	INPUT	EXPECTED OUTPUT	OBTAINED OUTPUT	PASS/FAIL	REMARKS
TC01		4.09	4.09	PASS	SUCCESS
TC02		3.14	3.14	PASS	SUPER SUCCESS
TC03		2.53	2.49	FAIL	FAILURE
TC04		0.45	0.45	PASS	SUCCESS

CHAPTER 9

CONCLUSION AND FUTURE ENHANCEMENT

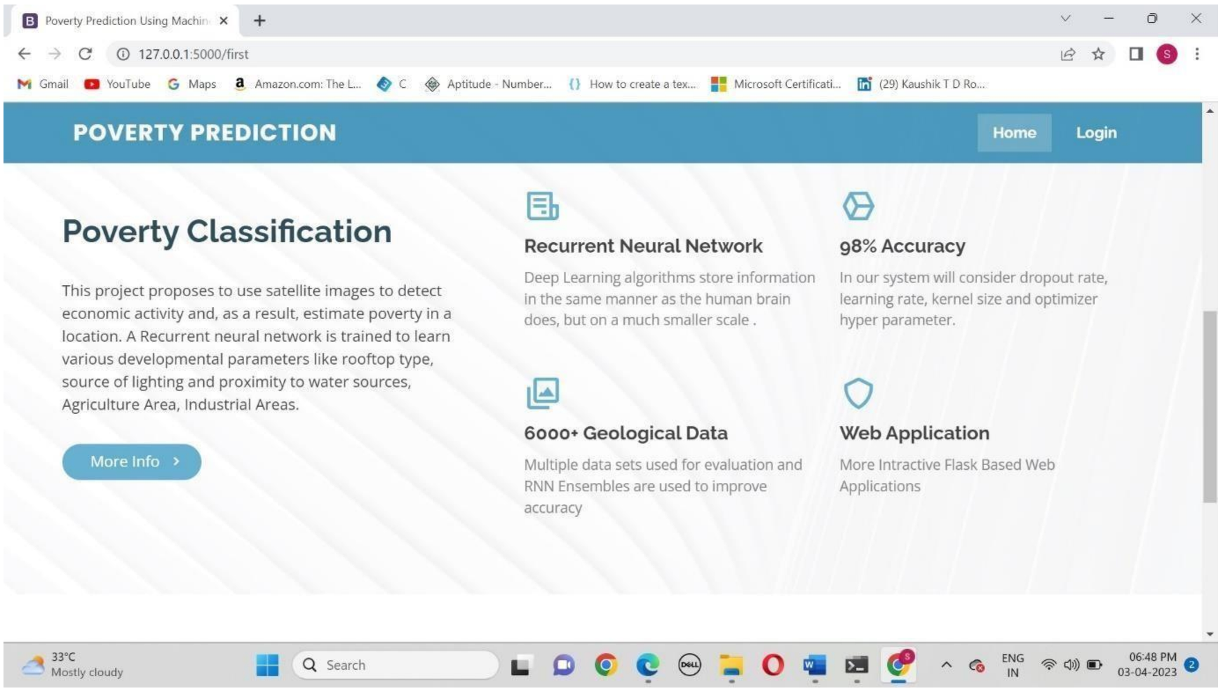
9.1 CONCLUSION

The goal of this Project was to examine a novel approach to poverty prediction using both daytime and night-time satellite images. While there is more to do, we think our work can provide a basis for poverty prediction as well as show its potential. Identifying regions of poverty is the first step on its reduction and believe this work has contributed.

9.2 FUTURE ENHANCEMENT

In future along with poverty level it will tell what wealth has to improve to get rid of poverty.

APPENDICES


Poverty Prediction Using Machin
127.0.0.1:5000/first

POVERTY PREDICTION

Home Login

Poverty Classification

This project proposes to use satellite images to detect economic activity and, as a result, estimate poverty in a location. A Recurrent neural network is trained to learn various developmental parameters like rooftop type, source of lighting and proximity to water sources, Agriculture Area, Industrial Areas.

More Info

Recurrent Neural Network

Deep Learning algorithms store information in the same manner as the human brain does, but on a much smaller scale .

98% Accuracy

In our system will consider dropout rate, learning rate, kernel size and optimizer hyper parameter.

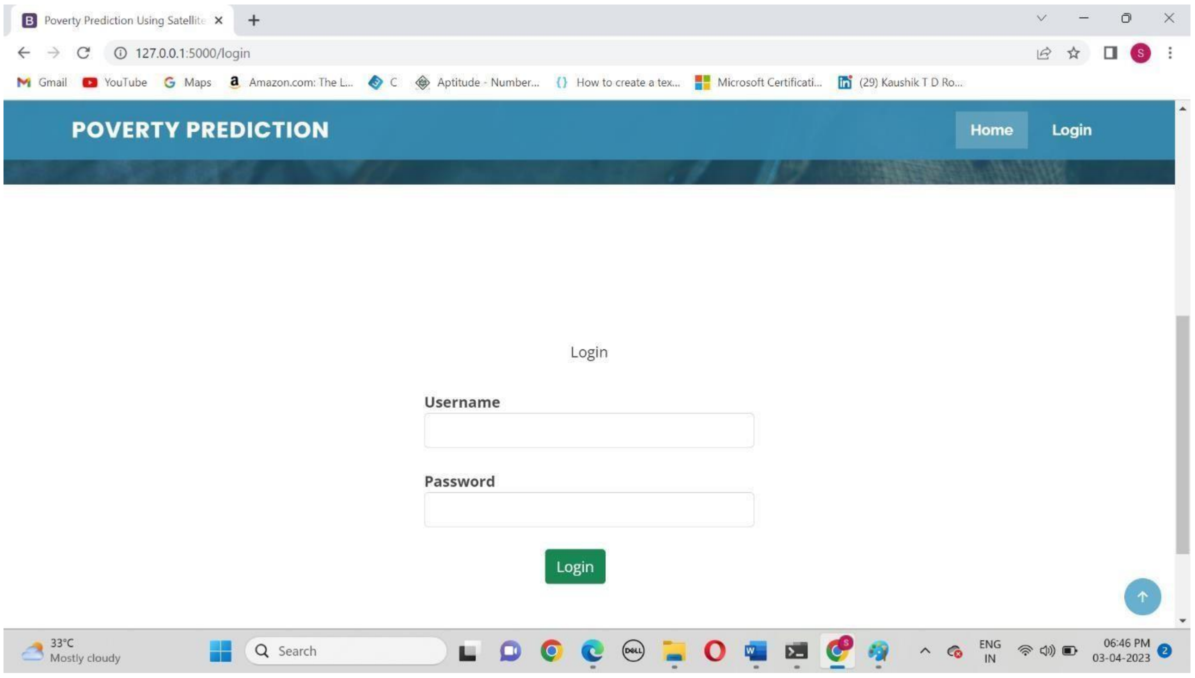
6000+ Geological Data

Multiple data sets used for evaluation and RNN Ensembles are used to improve accuracy

Web Application

More Intractive Flask Based Web Applications

Fig A.1 Home Page


Poverty Prediction Using Satellite
127.0.0.1:5000/login

POVERTY PREDICTION

Home Login

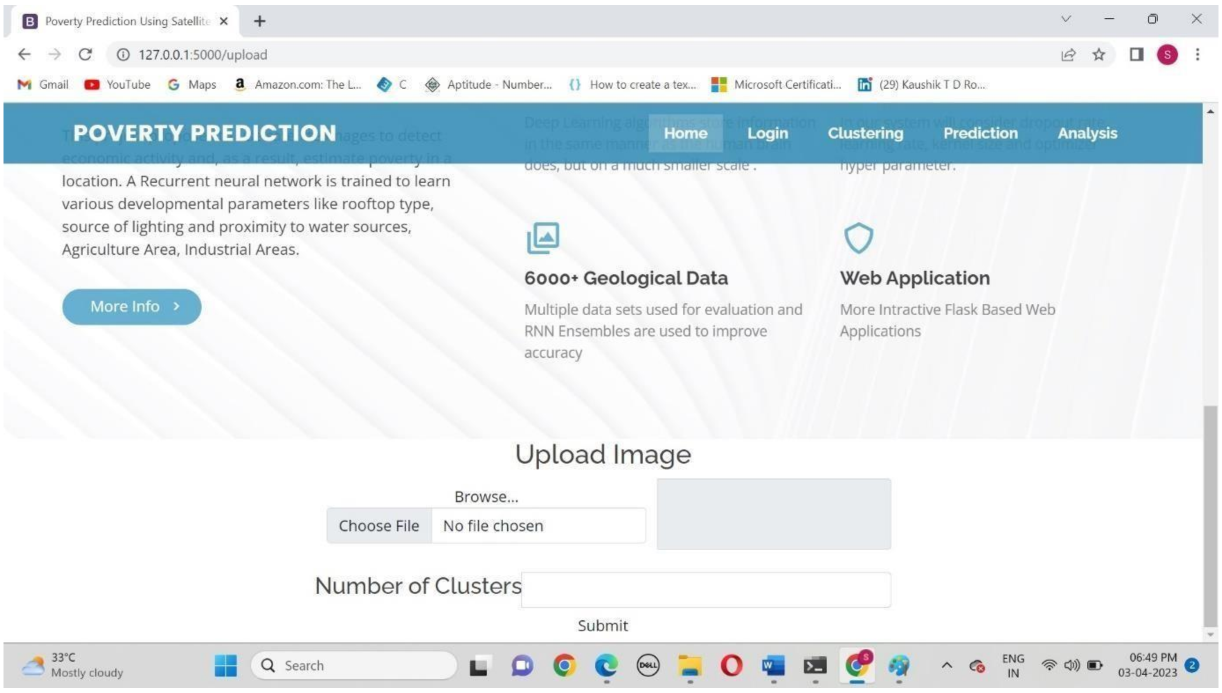
Login

Username

Password

Login

Fig A.2 Login Page



location. A Recurrent neural network is trained to learn various developmental parameters like rooftop type, source of lighting and proximity to water sources, Agriculture Area, Industrial Areas.

Multiple data sets used for evaluation and RNN Ensembles are used to improve accuracy

More Intractive Flask Based Web Applications

Fig A.3 Clustering page

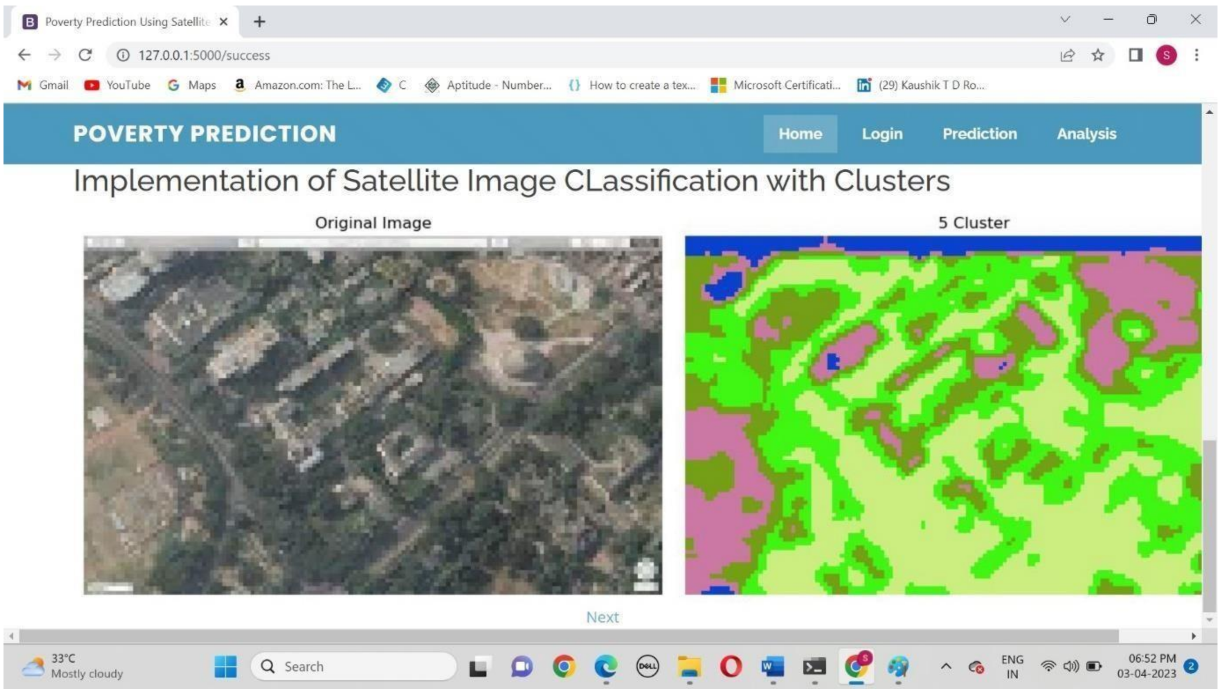


Fig A.4 Clustering the region

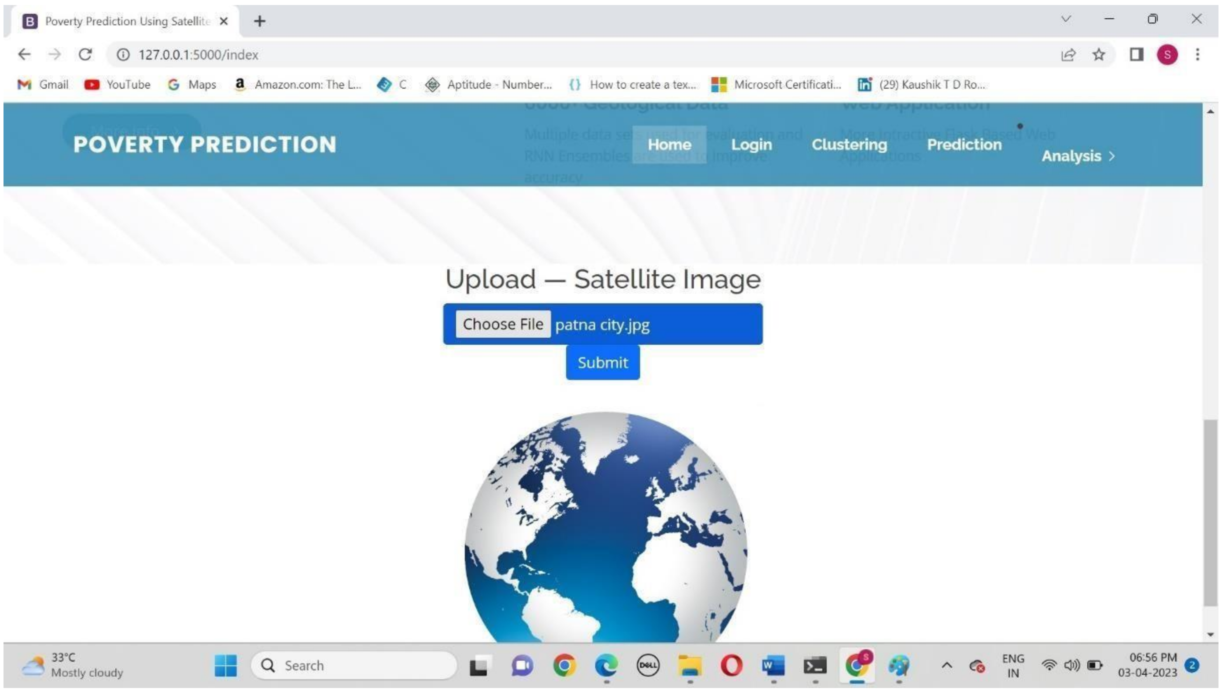
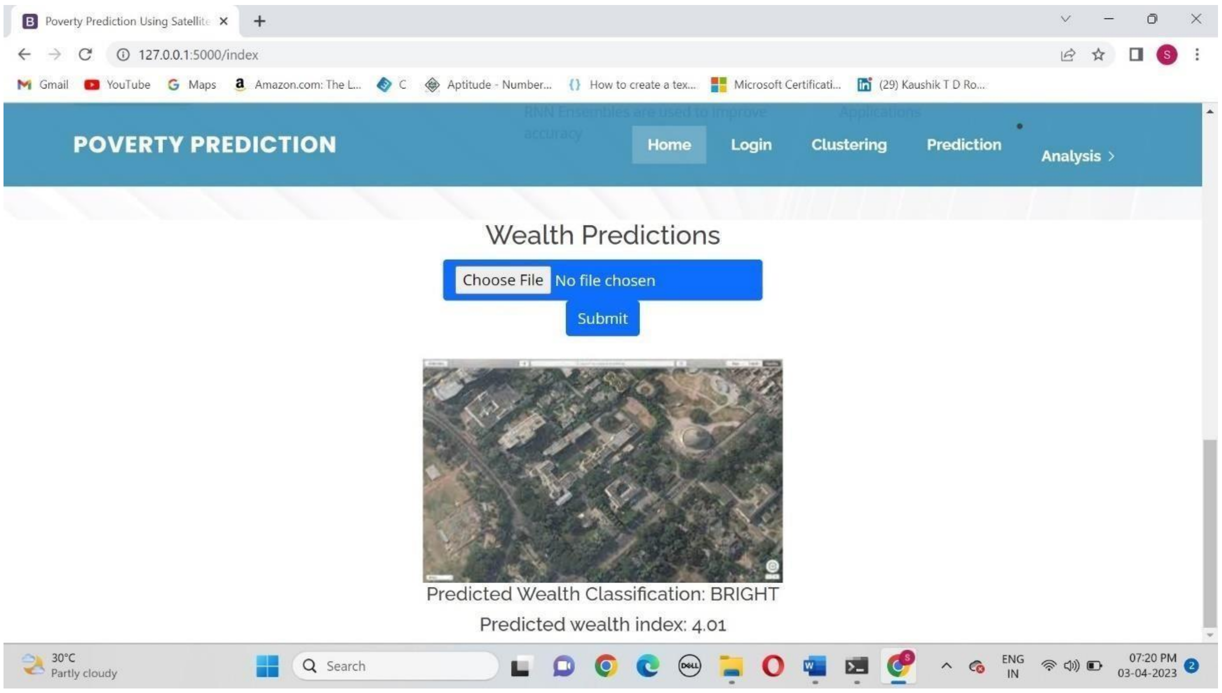


Fig A.5 Prediction Page



Predicted Wealth Classification: BRIGHT

Predicted wealth index: 4.01

Fig A.6 Wealth Prediction

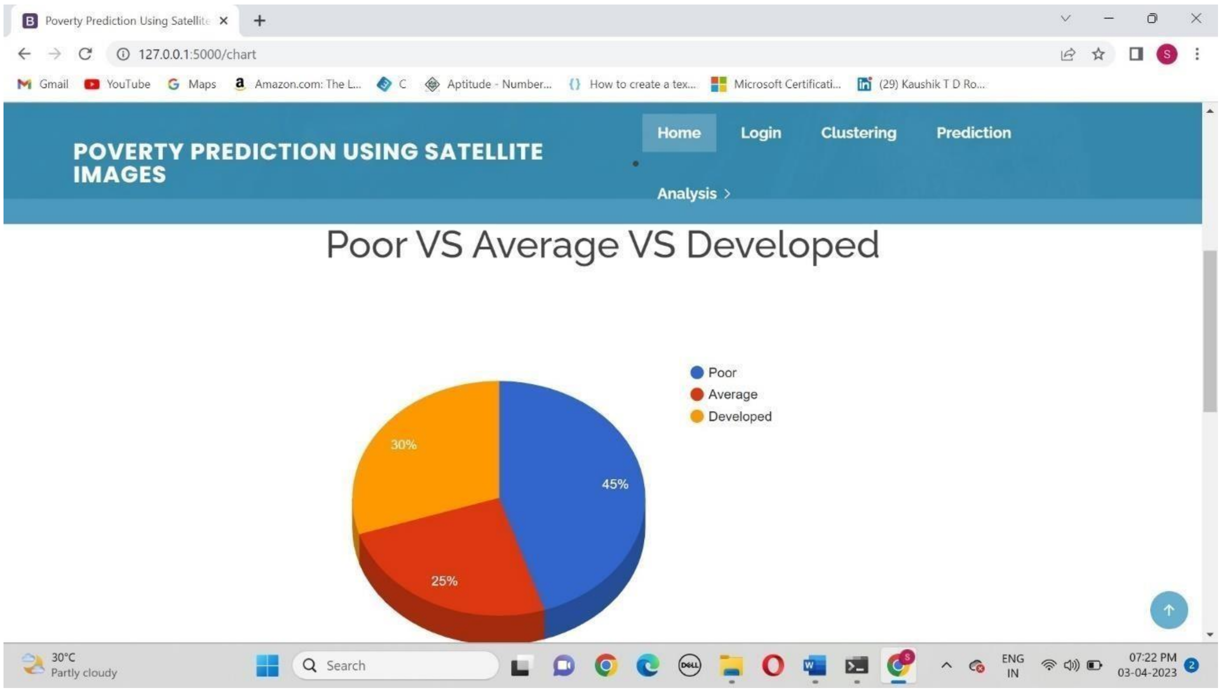

Poverty Prediction Using Satellite
127.0.0.1:5000/chart
POVERTY PREDICTION USING SATELLITE IMAGES
Home
Login
Clustering
Prediction
Analysis
Poor VS Average VS Developed
Poor
Average
Developed
45%
25%
30%

Fig A.7 Accuracy Pie Chart

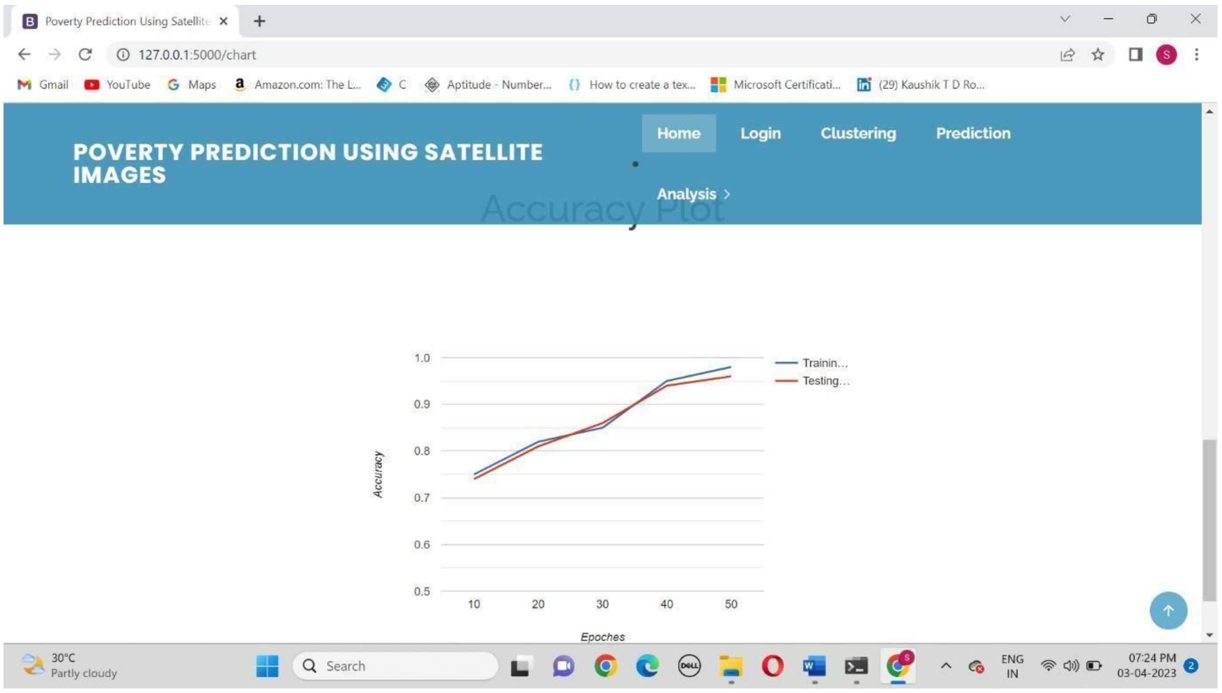


Fig A.8 Accuracy Graph

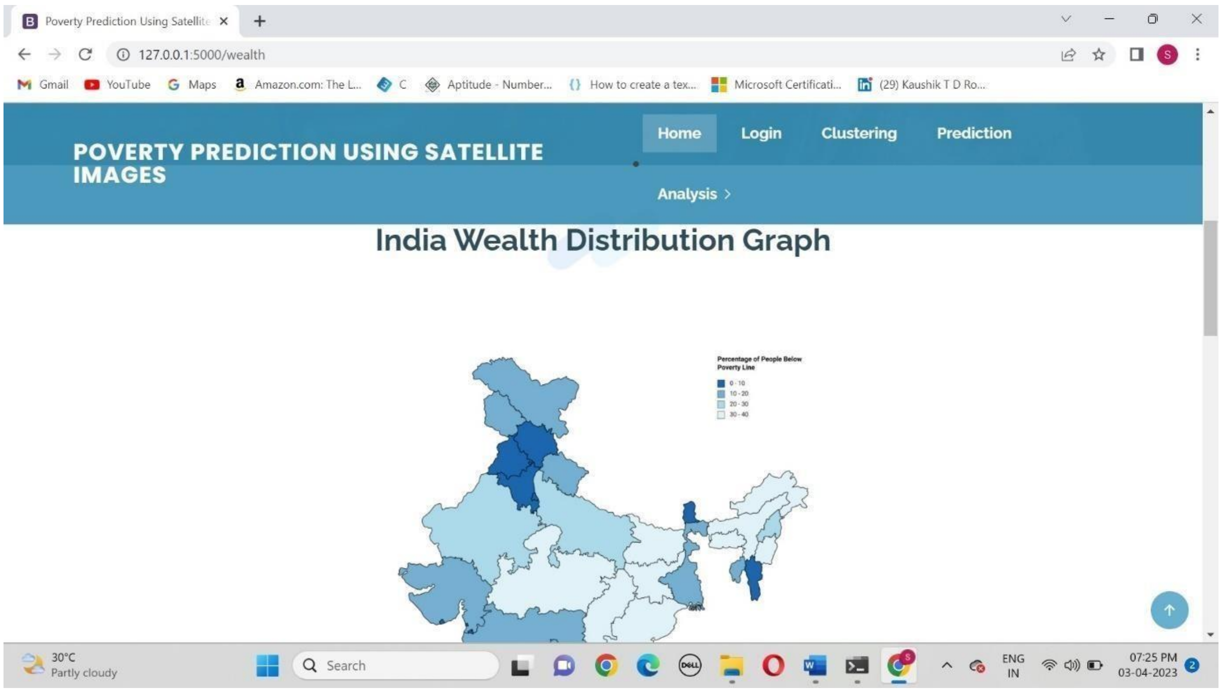

Poverty Prediction Using Satellite
127.0.0.1:5000/wealth
POVERTY PREDICTION USING SATELLITE IMAGES
Home
Login
Clustering
Prediction
Analysis
India Wealth Distribution Graph
Percentage of People Below Poverty Line
0 - 10
10 - 20
20 - 30
30 - 40

Fig A.9 Wealth Distribution graph-India

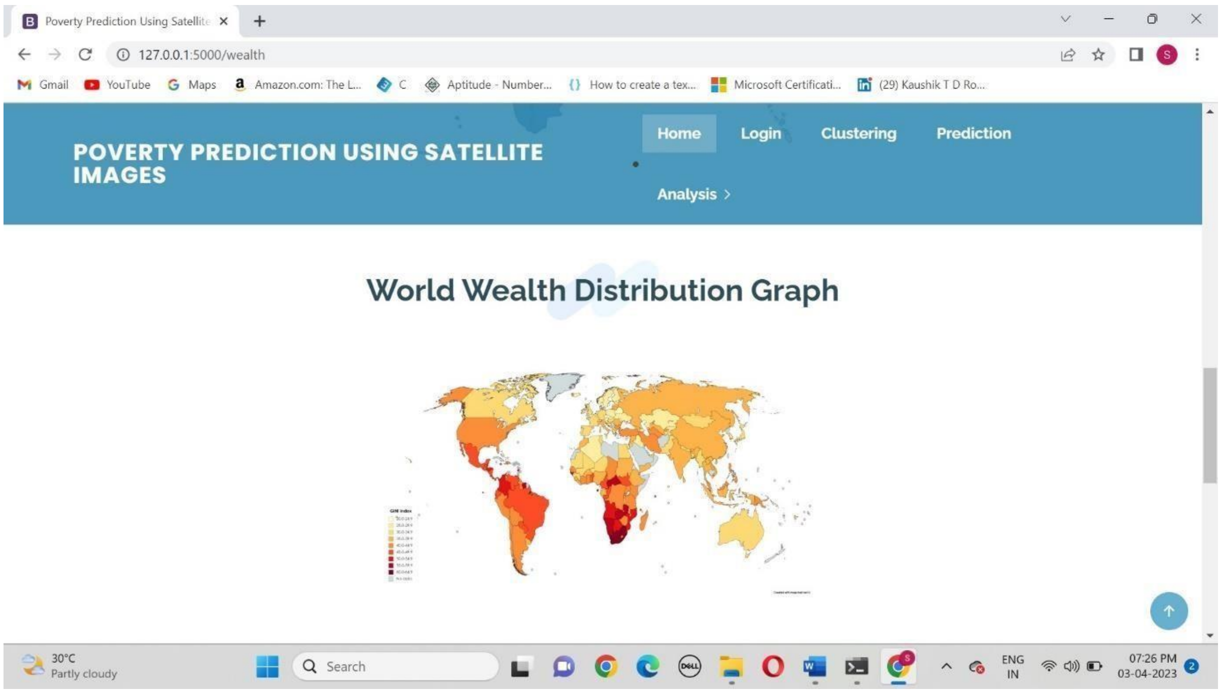

Poverty Prediction Using Satellite
127.0.0.1:5000/wealth
POVERTY PREDICTION USING SATELLITE IMAGES
Home
Login
Clustering
Prediction
Analysis
World Wealth Distribution Graph

Fig A.10 Wealth Distribution graph

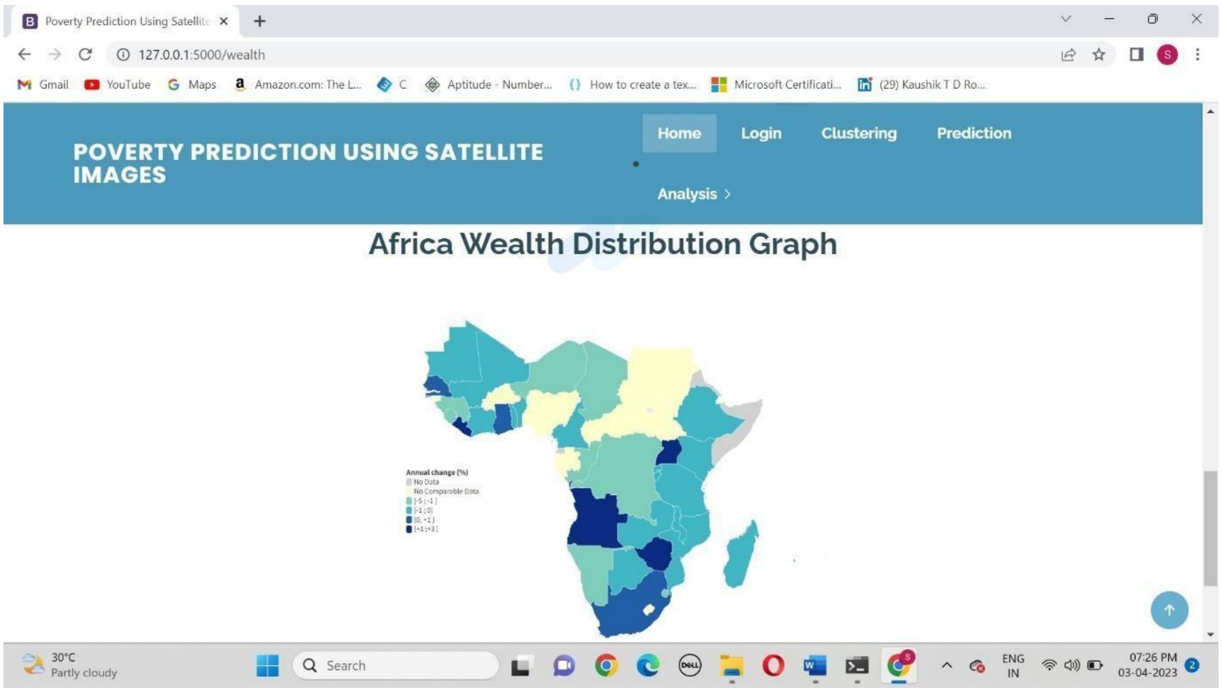


Fig A.11 Wealth Distribution graph-Africa

REFERENCES

[1] S. S. A. Zaidi, M. S. Ansari, A. Aslam, N. Kanwal, M. Asghar, and B. Lee. A survey of modern deep learning based object detection models, 2021.

[2] C. Yeh, A. Perez, A. Driscoll, G. Azzari, Z. Tang, D. Lobell, S. Ermon, and M. Burke. Using publicly available satellite imagery and deep learning to understand economic well-being in africa. Nat. Commun.,11(1):2583, May 2022.

[3] Ayush. Kumar, Uzkent. Burak, Burke. Marshall, Lobell. David and Stefano Ermon, "Generating Interpretable Poverty Maps using Object Detection in Satellite Images", 2023.

[4] Varun Chitturi , Zaid Nabulsi. Predicting Poverty Level from Satellite Imagery using Deep Neural Networks, 2022.

[5] Jonnadula Prasanna, Mounika Susarla, K. Suvarna Vani, Harsha Vardhan Govada.Prediction of Population Density & Poverty Rate Using Uncertain Mosaics with Satellite Imagery,2022.

[6]] W. H. Li, L.Wen, and Y. B. Chen, ``Application of improved GA-BP neural network model in property Poverty prediction," Geomatics Inf. Sci. Wuhan Univ., vol. 42, no. 8, pp. 11101116, 2021.

[7] J. M. Caplan, L. W. Kennedy, and J. Miller, ``Risk terrain modeling: Brokering criminological theory and GIS methods for Poverty forecasting," Justice Quart., vol. 28, no. 2, pp. 360381, 2021.

[8] M. Cahill and G. Mulligan, ``Using geographically weighted regression to explore local Poverty patterns," Social Sci. Comput. Rev., vol. 25, no. 2, pp. 174193, May 2021.

[9] A. Almehmadi, Z. Joudaki, and R. Jalali, ``Language usage on Twitter predicts Poverty rates," in Proc. 10th Int. Conf. Secur. Inf. Netw. (SIN), 2021.

[10]H. Berestycki and J.-P. Nadal, ``Self-organised critical hot spots of criminal activity," Eur. J. Appl. Math., vol. 21, nos. 45, pp. 371399, Oct. 2021.

[11] K. C. Baumgartner, S. Ferrari, and C. G. Salfati, ``Bayesian network modeling of offender behavior for criminal proling,'' in Proc. 44th IEEE Conf. Decis. Control, Eur. Control Conf. (CDC-ECC), Dec. 2022, pp. 27022709.

[12] W. Gorr and R. Harries, ``Introduction to Poverty forecasting,'' Int. J. Fore-casting, vol. 19, no. 4, pp. 551555, Oct. 2023.

[13] W. H. Li, L.Wen, and Y. B. Chen, ``Application of improved GA-BP neural network model in property Poverty prediction,'' Geomatics Inf. Sci. Wuhan Univ., vol. 42, no. 8, pp. 11101116, 2021.

[14]] L. Lin,W. J. Liu, andW.W. Liao, ``Comparison of random forest algorithm and space-time kernel density mapping for Poverty hotspot prediction,'' Prog. Geogr., vol. 37, no. 6, pp. 761771, 2022.

[15] C. L. X. Liu, S. H. Zhou, and C. Jiang, ``Spatial heterogeneity of microspatial factors' effects on street robberies: A case study of DP Peninsula,'' Geograph. Res., vol. 36, no. 12, pp. 24922504, 2021.