***Project Title: Global Earth Observation & Land Use Change Analysis***

***NOTE:* Satellite images are collected and analysed then required features are drawn from the images and presented here in the Excel sheet as Dataset\_GEOLUC as input for this project**

***SECTION 1: PROJECT INTRODUCTION***

**This project analyzes 10,000 records of Global regional land cover data (2000–2024) enriched with features like NDVI, rainfall, temperature anomalies, night-lights, and population density.**

**Using descriptive analytics for the above mentioned feature relationships, it reveals historical trends in deforestation, urbanization, and agricultural expansion.**

**Machine learning models are then applied to predict future land cover scenarios, and prescriptive insights are generated to support policy planning for sustainable land and resource management.**

**This project aims to leverage satellite imagery-derived indicators — such as NDVI, temperature anomalies, rainfall, night-time lights, and population density — along with machine learning techniques to analyze patterns, detect trends, and predict future land cover changes across the globe from 2000 to 2024.**

***SECTION 2: PROJECT PROBLEM STATEMENT***

* **Quantify historical land cover transitions (forest loss, urban growth, etc.) by region.**
* **Identify key drivers of land use change using feature importance analysis.**
* **Predict future land cover scenarios using deep learning models.**
* **Support decision-making by generating prescriptive insights, such as priority zones for afforestation, water resource management, or urban planning interventions.**

***SECTION 3: PROJECT OBJECTIVE***

**The primary objective of this project is to analyze, model, and predict land use and land cover (LULC) changes across Global using a combination of satellite imagery-derived indicators and machine learning techniques.**

**By doing so, we aim to provide actionable insights that help the below end users to make data-driven decisions for sustainable land management.**

1. **policymakers,**
2. **urban planners, and**
3. **environmental agencies**

***SECTION 4: PROJECT END RESULT\_PRESCRIPTION / RECOMMENDATIONS TO END USERS***

* + **Use model outputs to recommend interventions — e.g., regions requiring reforestation, smart urban planning zones, or water conservation measures.**
  + **Simulate “what-if” scenarios for policy planning (e.g., impact of reducing urban expansion by 10%).**

***SECTION 5: SCOPE OF THE PROJECT***

1. **Data Exploration & Preprocessing**
   * **Clean, transform, and validate the uploaded dataset containing 10,000 records of India’s regional land cover, NDVI, rainfall, population density, temperature anomalies, and night-light indices.**
   * **Aggregate and segment data region-wise and year-wise to enable time-series and spatial analysis.**
2. **Descriptive Analytics**
   * **Generate visualizations (heatmaps, trend charts, choropleth maps) to show spatial and temporal patterns in land cover changes.**
   * **Identify major hotspots of deforestation, urban expansion, and water body depletion.**

1. **Predictive Modeling**
   * **Build and train a machine learning model (e.g., using Random Forest for classification type prediction i.e for categorical target values whose values are discrete labels and XGBoost for continuous target numerical values) to forecast future LULC distributions.**
   * **Evaluate performance using metrics such as accuracy, F1-score, and RMSE.**
2. **Prescriptive Insights - End Result of this Project**
   * **Use model outputs to recommend interventions — e.g., regions requiring reforestation, smart urban planning zones, or water conservation measures.**
   * **Simulate “what-if” scenarios for policy planning (e.g., impact of reducing urban expansion by 10%).**
3. **Stakeholder-Friendly Dashboard**
   * **Design a Power BI dashboard or web-based visualization with filters by region, land cover type, and year.**
   * **Provide interactive charts, maps, and tables with predictive trends and actionable recommendations.**

***SECTION 6: PROJECT DELIVERABLES***

**✅ Cleaned & Documented Dataset – Preprocessed, analysis-ready CSV file with metadata.  
 ✅ Exploratory Data Analysis (EDA) Report – Charts, correlation matrices, and summary statistics.  
 ✅ Deep Learning Model & Code – Well-documented code for LULC prediction with reproducible results.  
 ✅ Feature Importance & Driver Analysis – Ranked list of factors influencing land cover change.  
 ✅ Scenario-Based Recommendations – Policy suggestions based on model outputs.  
 ✅ Interactive Dashboard – Power BI / web dashboard to communicate findings to decision-makers.**

***SECTION 7: TARGET VARIABLES: Land\_Cover\_Type & Area\_sq\_km***

**Why below 2 features are selected as the target variable choices?**

**1.** **For Land\_Cover\_Type target variable classification type prediction will be used as it is a natural predictive task for land-use analysis — it helps assign land-cover labels to new or unlabeled observations (e.g., vector parcels produced from image segmentation).**

**🔹 Land\_Cover\_Type target variable is categorical type with different classes and will be used for Classification type prediction**

* **Target Feature (Y): Categorical (discrete labels)**
  + **Examples: *Yes/No, Pass/Fail, Healthy/Sick, Crop Type (Wheat/Rice/Maize)*.**
  + **Can be binary classification (two classes) or multi-class classification (three or more classes).**
* **Goal: Predict which class/label an observation belongs to.**
* **Algorithms that can be used: Random Forest, XGBoost (Classifier),**
* **Evaluation metrics: Accuracy, Precision, Recall, F1-score, AUC-ROC.**

**✅ When to choose Classification type prediction: If your target column is categorical (nominal or ordinal), you’re doing classification.**

**2.**  **For target variable** **Area\_sq\_km regression type prediction is useful to estimate or gap-fill missing area measurements and for change-quantification tasks.**

**🔹 Area\_sq\_km target variable is continuous target numerical values and will be used for Regression type prediction**

* **·Target Feature (Y): Continuous (numeric values)**
  + **Examples: *House Price, Rainfall Amount, Land Area (sq.km), Population Density, Crop Yield (tons)*.**
* **Goal: Predict a numerical value as closely as possible to the actual.**
* **Algorithms that can be used: Linear Regression, Random Forest Regressor, XGBoost (Regressor), Gradient Boosting, etc.**
* **Evaluation metrics: RMSE, MAE, R², MAPE.**

**✅ When to choose Regression type prediction: If your target column is continuous (real numbers), you’re doing regression.**

***SECTION 8: DICTIONARY FOR DATASET***

| **Column Name** | **Data Type** | **Description** | **Example Values** |
| --- | --- | --- | --- |
| **Region** | Categorical | Geographic region where the observation was recorded. Covers major continents or zones. | North America, Africa, Asia, Arctic |
| **Year** | Integer | Year of observation (2000–2024). Used for trend and time-series analysis. | 2005, 2015, 2023 |
| **Land\_Cover\_Type** | Categorical | Type of land cover being observed. | Forest, Agriculture, Urban, Water, Barren Land, Wetlands |
| **Area\_sq\_km** | Float | Total area (in square kilometers) covered by this land type for the given region and year. | 5123.45, 18560.33 |
| **NDVI** | Float (0.0 – 1.0) | Normalized Difference Vegetation Index – measures vegetation health (higher = denser, greener vegetation). | 0.15 (Sparse), 0.85 (Very dense) |
| **Population\_Density** | Float | Population per square kilometer in that region for the observed year. | 45.2, 1200.5 |
| **Temperature\_Anomaly** | Float | Deviation in temperature (°C) from a baseline climatological average. Positive = warmer than usual, negative = cooler. | +1.2, -0.8 |

***SECTION 9: STEPS TO SHOWCASE DESCRIPTIVE ANALYTICS IN THE FRONTEND USING STREAMLIT APP (PYTHON LIBRARY) - IN LOCAL MACHINE***

**Quick steps to run Streamlit on a local machine**

1. **STEP 1: First create the streamlit\_app.py file in notepad and then save this file in the path c:\users\GVT\\GEOLUC\ or in any other path of your choice (use the same path while executing in the power shell or in command prompt)**

**(this notepad text file streamlit\_app.py is given in the folder shared with you as streamlit\_app.txt, just save this file as streamlit\_app.py using notepad in the C folder path like “C:\users\GVT\GEOLUC\” - instead of GVT you create your name folder and create folder GEOLUC in your system C path as given above)**

**(streamlit\_app.py file has python code for GEOLUC project descriptive analytics along with streamlit code to show descriptive analytics in the front end in the browser of a local machine)**

1. **STEP 2: Open PowerShell (Start → type PowerShell → Enter).  
   If you prefer, use PowerShell (Admin) for installation steps that need permissions.**
2. **STEP 3: Change to the folder where you saved streamlit\_app.py:**

**cd "C:\Users\GVT\GEOLUC"**

1. **STEP 4:  *(Optional but recommended)* Create and activate a virtual environment (run the below commands one by one)**

**python -m venv venv**

**# To Activate venv in PowerShell:(don’t execute this its just a comment)**

**.\venv\Scripts\Activate.ps1**

**If activation fails due to execution policy, run once (as yourself — not admin necessarily): (run the below commands one by one)**

**Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser -Force**

**.\venv\Scripts\Activate.ps1**

1. **STEP 5: Upgrade pip and install required packages (run the below commands one by one)**

**python -m pip install --upgrade pip**

**pip install streamlit pandas plotly numpy**

1. **STEP 6: Run the Streamlit app (this command will keep running and show logs):**

**streamlit run streamlit\_app.py**

**If streamlit isn't recognized, run:**

**python -m streamlit run streamlit\_app.py**

1. **STEP 7: If automatically the browser doesn't open then Open your browser to:**  <http://localhost:8501>

**(or click the Local URL printed by Streamlit in the terminal).**

***SECTION 10: STEPS TO SHOWCASE DESCRIPTIVE ANALYTICS IN THE FRONTEND FOR PUBLIC USERS USING GIT REPO AND STREAMLIT CLOUD - Used in Production phase or Deployment phase of the project***

**Steps to Deploy on Streamlit Community Cloud (no tunneling is required here)**

**(Use this If you want a public app without local setup: as you did in the above step of running streamlit in local machine)**

1. Create a GitHub repo and push streamlit\_app.py to it (and requirements.txt with streamlit,pandas,plotly,numpy).
2. Go to https://share.streamlit.io, sign in with GitHub, choose the repo and branch, and deploy.
3. Streamlit Cloud gives you a public URL; no tunneling or tokens required.

**Steps to deploy the GEOLUC Project descriptive analytics streamlit output on the streamlit cloud for free using Github repo**

Deploying your **streamlit\_app.py** to Streamlit Community Cloud (share.streamlit.io) is a great choice — it's free for public repos and super simple. Below is the compact, step-by-step guide with exact files what you should create - the requirements.txt content, and the Git commands to push your app to GitHub and then deploy on Streamlit Cloud.

How it works (short)

1. Put streamlit\_app.py and a small requirements.txt into a Git repo.
2. Push the repo to GitHub.
3. Sign in at https://share.streamlit.io and connect that GitHub repo — Streamlit Cloud builds and serves the app and gives you a public URL.

Files you should have in the repo

* streamlit\_app.py ← your app (the file you already created)
* requirements.txt ← packages Streamlit Cloud will install
* optional: README.md, .gitignore, any small data files (avoid very large datasets; use external storage or Git LFS)

**Steps to push to Git and deploy in Streamlit cloud**

**Step A: Push the repo to GitHub (public).**

**Step B: Deploy in Streamlit community cloud**

* 1. **Go to Streamlit Community Cloud.**
  2. **Select your repo → branch → streamlit\_app.py.**
  3. **Deploy! 🚀**

**Below are the detailed steps for the above mentioned STEP A :Push the repo to GitHub (public).**

**Step 1: Create repo locally → push → deploy**

**A. Create a new folder and copy files**

1. Put streamlit\_app.py into a new folder, e.g. GEOLUC-streamlit/.
2. Create requirements.txt in the same folder (contents given below).
3. Optional: create README.md (contents given below)

**B. Commit and push to GitHub**

Open a terminal (PowerShell on Windows, Terminal on Mac/Linux) and run these commands inside the folder (cd path\to\GEOLUC-streamlit):

If you already have a GitHub repo and remote set up, skip the git remote add step and just push.

**Windows / macOS / Linux (copy-paste one block — adapt your GitHub repo URL):**

# initialize git (if not already)

git init

git add .

git commit -m "Initial commit: Streamlit GEOLUC descriptive EDA app"

# create a new repo on GitHub (manual option below)

# replace <USERNAME> and <REPO> with your GitHub username and desired repo name

git remote add origin https://github.com/<USERNAME>/<REPO>.git

# push to GitHub main branch (may be 'main' or 'master' depending on your git config)

git branch -M main

git push -u origin main

**Note: If you prefer to create the repo via GitHub web UI:**

* Go to <https://github.com> → New Repository → name it (e.g. geoluc-streamlit), make it Public (or Private if you have Streamlit Pro), then GitHub shows the git remote add & git push commands you can copy to your terminal.

**Step 2: Minimal requirements.txt (copy this)**

streamlit

pandas

plotly

numpy

(You can pin versions like streamlit==1.25.0 if you need a specific version.)

**Step 3: (Optional) : 📄 README.md**

# GEOLUC Descriptive EDA Streamlit App

This is an interactive \*\*Streamlit\*\* app for exploring \*\*Global Earth Observation & Land Use Change (GEOLUC)\*\* data.

It provides 12 descriptive analytics charts with automated textual insights, allowing end users to filter by \*\*Region, Land Cover Type, and Year Range\*\*.

## Features

- 📊 Interactive charts (bar, line, stacked area, histogram, scatter, heatmap, boxplots)

- 🔎 Dynamic filtering (region, land cover type, years)

- 📝 Automated textual insights under each chart

- 📂 Upload your own CSV (expected columns: `Region`, `Land\_Cover\_Type`, `Year`, `Area\_sq\_km`, `NDVI`, `Population\_Density`, `Temperature\_Anomaly`)

## Running locally

1. Clone this repo:

```bash

git clone https://github.com/<YOUR\_USERNAME>/<YOUR\_REPO>.git

cd <YOUR\_REPO>

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Here ends the Readme file contents\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Step 4: (Optional) : Create a virtual environment:**

python -m venv venv

.\venv\Scripts\activate # On Windows

**Step 5: Install requirements:**

pip install -r requirements.txt

**Step 6: Run the app:**

streamlit run streamlit\_app.py

**Step 7**: Open <http://localhost:8501> in your browser.

**Below are the detailed steps for the above mentioned STEP B : Deploy on Streamlit Community Cloud**

**Step 1:**  Go to https://share.streamlit.io and sign in with GitHub.

**Step 2:** Click **“New app”** (top-right) → choose your GitHub repo, branch (usually main), and the file path to streamlit\_app.py.

**Step 3:** Click **Deploy**. Streamlit Cloud will install packages from requirements.txt, build, and start your app.

**Step 4:** When done, you’ll get a public URL like https://<your-app>.streamlit.app.

**Note: place \*\*both files\*\* in the same folder as `streamlit\_app.py`, commit, and push to your GitHub repo.**

**NOTE: Helpful tips & common gotchas**

* **Large datasets**: don’t store huge files in the repo. Instead:
  + host the CSV on Google Drive / AWS S3 / GitHub release and have the app download it at runtime, or
  + use the Streamlit file\_uploader to let users upload CSVs from their machine.
* **Private repos**: free Streamlit Cloud allows only public repos; private repo support requires a paid plan or using your personal account settings. (Check current Streamlit policies.)
* **Python versions**: Streamlit Cloud uses default Python. If you need a specific runtime, add runtime.txt (rarely needed). Most apps work without it.
* **Secrets (API keys)**: store them under the app’s **Settings → Secrets** in Streamlit Cloud — then access via st.secrets["MY\_KEY"].
* **Requirements**: pin package versions if you want consistent builds; otherwise omit pins to use latest.
* **Automatic redeploy**: pushing commits to the GitHub branch automatically redeploys the app.
* **Logs and errors**: the Streamlit Cloud dashboard shows build logs and runtime logs if something fails — use them to debug missing packages or errors.

***SECTION 11: Ensure whether you have crosschecked for the below mentioned file paths before executing Predictive analytics python code - “DA Pjt\_GEOLUC\_Predictive.ipynb” file in the google colab***

***Very Important :***

***Point 1: To run Predictive Analytics python code - “DA Pjt\_GEOLUC\_Predictive.ipynb” file in google colab ensure the dataset - “Global 2\_Land\_Use\_Change\_10000.csv” file is in the same path what you have given in your file “DA Pjt\_GEOLUC\_Predictive.ipynb”***

***Point 2: After executing the predictive analytics code - “DA Pjt\_GEOLUC\_Predictive.ipynb” file in google colab, you get the below output files, check whether these files are generated in the same path as what is given in your code i.e /content/file name, check for these output files in google colab file explorer***

1. ***Predictions\_classification.csv***
2. ***Predictions\_regression.csv***
3. ***Xgb\_classification\_model.joblib***
4. ***Xgb\_regression\_model.joblib***
5. ***feature\_importances\_classification.csv***

***SECTION 12: STEPS TO SHOWCASE PREDICTIVE ANALYTICS IN THE FRONTEND USING STREAMLIT APP (PYTHON LIBRARY) - IN LOCAL MACHINE***

**🚀 How to Run in PowerShell**

**STEP 1: First create the train\_geoluc.py file in notepad and then save this file in the path c:\users\GVT\\GEOLUC\ or in any other path of your choice (use the same path while executing in the power shell or in command prompt)**

**(this notepad text file train\_geoluc.py is given in the folder shared with you as train\_geoluc.txt, just save this file as train\_geoluc.py using notepad in the C folder path like “C:\users\GVT\GEOLUC\” - instead of GVT you create your name folder and create folder GEOLUC in your system C path as given above)**

**(train\_geoluc.py file has python code for GEOLUC project predictive analytics along with streamlit code to show predictive analytics in the front end in the browser of a local machine)**

**STEP 2: Save the file above as train\_geoluc.py (which is your predictive analytics code using streamlit app) in your project folder.(as you did for descriptive analytics in the above step - SECTION 9**

**STEP 3: Install dependencies if not already: (Run the below code)**

**pip install streamlit scikit-learn xgboost shap matplotlib seaborn pandas**

**STEP 4: Install missing dependency (Run below code one by one)**

**.\venv\Scripts\activate**

**pip install joblib shap xgboost scikit-learn matplotlib pandas**

**python train\_geoluc.py**

**STEP 5: Run the app: (Run the below code)**

**streamlit run train\_geoluc.py**

**STEP 6: Open the local server link shown (default: http://localhost:8501) in your browser.**

***SECTION 13: STEPS TO SHOWCASE PREDICTIVE ANALYTICS IN THE FRONTEND FOR PUBLIC USERS USING GTI REPO AND STREAMLIT CLOUD - Used in Production phase or Deployment phase of the project***

***Follow the same steps given above in SECTION 10 but instead of streamlit\_app.py file use train\_geoluc.py file***

***SECTION 14: Ensure you have executed the predictive analytics python code before you run the prescriptive analytics python code and also check whether you have the below mentioned file paths before executing Prescriptive analytics python code - “DA Pjt\_GEOLUC\_Prescriptive.ipynb” file in the google colab***

***Very Important :***

***Point 1: First you should run the Predictive analytics python code - “DA Pjt\_GEOLUC\_Predictive.ipynb” in google colab before you run Prescriptive Analytics python code - “DA Pjt\_GEOLUC\_Prescriptive.ipynb” , because the output files from predictive analytics is used as input files for prescriptive analytics code***

***Point 2: Before you run Prescriptive Analytics python code - “DA Pjt\_GEOLUC\_Prescriptive.ipynb” file in google colab, you ensure the below input files (which are actually the output files from “DA Pjt\_GEOLUC\_Prescriptive.ipynb” file) are available in the same path in the google drive as you have mentioned in “DA Pjt\_GEOLUC\_Prescriptive.ipynb” file***

***Hence check for the presence of below mentioned files in the same paths as given in the below snippet of code (present in “DA Pjt\_GEOLUC\_Prescriptive.ipynb” file)***

***classification\_preds\_path = "/content/drive/MyDrive/predictions\_classification.csv"***

***regression\_preds\_path = "/content/drive/MyDrive/predictions\_regression.csv"***

***original\_data\_path = "/content/drive/MyDrive/Global 2\_Land\_Use\_Change\_10000.csv"***

***Note :***

1. ***predictions\_classification.csv and predictions\_regression.csv are the output files from python code-“DA Pjt\_GEOLUC\_Predictive.ipynb” file***
2. ***In the above mentioned path - /content is nothing but your google drive path /drive/MyDrive but you can check this in your pythoncode - “DA Pjt\_GEOLUC\_Predictive.ipynb” file explorer present in left side pane.***

***SECTION 15: STEPS TO SHOWCASE PRECRIPTIVE ANALYTICS IN THE FRONTEND USING STREAMLIT APP (PYTHON LIBRARY) - IN LOCAL MACHINE***

**🚀 How to Run in PowerShell**

**STEP 1: First create the prescriptive\_app.py file in notepad and then save this file in the path c:\users\GVT\\GEOLUC\ or in any other path of your choice (use the same path while executing in the power shell or in command prompt)**

**(this notepad text file prescriptive\_app.py file is given in the folder shared with you as prescriptive\_app.txt file just save this file as prescriptive\_app.py file using notepad in the C folder path like “C:\users\GVT\GEOLUC\” - instead of GVT you create your name folder and create folder GEOLUC in your system C path as given above)**

**prescriptive\_app.py file has python code for GEOLUC project prescriptive analytics along with streamlit code to show prescriptive analytics in the front end in the browser of a local machine)**

**STEP 2: Save the file above as a prescriptive\_app.py file (which is your predictive analytics code using streamlit app) in your project folder.(as you did for descriptive analytics in the above step - SECTION 9**

**STEP 3: Install required packages in your virtual environment or global Python: Run the below command in the power shell**

**pip install streamlit pandas numpy matplotlib seaborn**

**STEP 4: Also install (if not already installed): Run the below command in the power shell**

**pip install joblib**

**STEP 5 : Launch the App in PowerShell**

1. **Run Streamlit: Run the below command in the power shell**

**streamlit run prescriptive\_app.py**

1. **Your default browser will open at:** [**http://localhost:8501**](http://localhost:8501)
2. **When the browser opens In the front end it will ask you to upload the below files**
3. ***Input Dataset: Global 2\_Land\_Use\_Change\_10000.csv***
4. ***Output file from Predictive analytics python code : predictions\_classification.csv***

**Note : (Dataset might be in your MyDrive and predictions\_classification.csv is the output file from your predictive analytics python code when run in google colab and this might be your /content path as explained in Section 12)**

### **Note : You can now explore the prescriptive insights in a simple interactive dashboard. 🚀**

***SECTION 16: STEPS TO SHOWCASE PRESCRIPTIVE ANALYTICS IN THE FRONTEND FOR PUBLIC USERS USING GTI REPO AND STREAMLIT CLOUD - Used in Production phase or Deployment phase of the project***

***Follow the same steps given above in SECTION 10 but instead of streamlit\_app.py file use prescriptive\_app.py file***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***