|  |  |
| --- | --- |
| A picture of a winding road and trees  Telangana Forest Analysis  Using Data Cubes | Abstract  Unleash the Power of Geospatial Analysis: Explore, Analyze, and Visualize Satellite Imagery with Ease  Karnati Sai Krishna  Project School Documentation |

Project Report

**Project Title:** Telangana Forest Analysis using Data Cubes

**Project Members:**

K. Sai Krishna - 21BD1A050K

M. Venkata Varsha - 21BD1A050X

M. Sri Tharun - 21BD1A050V

N. Sumanth - 21BD1A0515

T. Jayadeep - 21BD1A051W

**Mentor:** Ram Mohan

**Domain:** Remote sensing

**Objective:** Develop a datacube using satellite images for analysis.

**Technologies:** Open Data Cube, QGIS, Flask, Machine Learning Algorithms

**Output:** The expected final product is a functional platform that can provide users with accurate and timely information about various changes in a region.

Contents

[1 Objectives 4](#_Toc140585022)

[2 Project Overview 5](#_Toc140585023)

[3 Introduction 7](#_Toc140585024)

[3.1 Data Cube 7](#_Toc140585025)

[3.2 Satellite Images and their Analysis 8](#_Toc140585026)

[3.2.1 Sentinel-2 8](#_Toc140585027)

[3.2.2 Landsat 9](#_Toc140585028)

[3.3 Open Data Cube 9](#_Toc140585029)

[3.4 Time Series Analysis 10](#_Toc140585030)

[3.5 Analysis Ready Data 10](#_Toc140585031)

[3.6 System Workflow 11](#_Toc140585032)

[3.7 Sen2Cor 12](#_Toc140585033)

[4 Setting up the Data Cube 14](#_Toc140585034)

[4.1 Data Cube Setup in Ubuntu Operating System 14](#_Toc140585035)

[4.2 Data Cube Setup in Docker Container 15](#_Toc140585036)

[5 Analysis through QGIS 16](#_Toc140585037)

[5.1 Installing QGIS 16](#_Toc140585038)

[5.2 Load Satellite Images 16](#_Toc140585039)

[5.3 Image Analysis 16](#_Toc140585040)

[5.4 Visualization and Output 18](#_Toc140585041)

[6 Analysis using Manual Method 19](#_Toc140585042)

[6.1 Download and Index Satellite Data 19](#_Toc140585043)

[6.2 Ingest the data locally 21](#_Toc140585044)

[6.3 Install Required Libraries 21](#_Toc140585045)

[6.4 Load Satellite Data 21](#_Toc140585046)

[6.5 Preprocess the Data 22](#_Toc140585047)

[6.6 Visualize Results and Changes 23](#_Toc140585048)

[7 Web Application 25](#_Toc140585049)

[7.1 Frontend – User Interaction 25](#_Toc140585050)

[7.2 Backend – Flask API 26](#_Toc140585051)

[8 Results 27](#_Toc140585052)

[8.1 Analysis Results 27](#_Toc140585053)

[8.2 Visualizations 28](#_Toc140585054)

[8.3 Data Interpretation 28](#_Toc140585055)

[8.4 User Experience 29](#_Toc140585056)

[8.5 Performance and Scalability 29](#_Toc140585057)

[9 Future Scope 30](#_Toc140585058)

[9.1 Additional Analysis Types 30](#_Toc140585059)

[9.2 Enhanced Visualization 30](#_Toc140585060)

[9.3 User-Driven Customization 30](#_Toc140585061)

[9.4 Performance Optimization 31](#_Toc140585062)

[9.5 Integration with External Data Sources 31](#_Toc140585063)

[9.6 User Authentication and Access Control 31](#_Toc140585064)

[10 References 32](#_Toc140585065)

# Objectives

Telangana is full of flora and fauna. In order to maintain its integrity, we need a proper tool to efficiently monitor the changes which are taking place in the forest areas. Our project is to develop a web application with a user interface that provides the detailed analysis of the forest cover in different regions and a monitoring tool for the vegetation in Telangana. The objectives of our project are:

**Forest change Detection and Monitoring:** An additional objective of the project is to develop tools and methodologies for timely forest change detection and monitoring using remote sensing data. By analysing the various bands data which is collected by satellites, the project aims to provide a clear change detection over a period of time by using various indices, assist in forest management efforts, and contribute to the reduction of deforestation in forest areas.

**Future Forest Cover Prediction:** The project also incorporates machine learning algorithm to predict forest cover change in the future. By leveraging historical satellite imagery data and other relevant variables, the project aims to develop models that can forecast changes in forest cover over time. This objective seeks to enhance our understanding of future forest dynamics, inform land-use planning decisions, and support proactive forest conservation strategies.

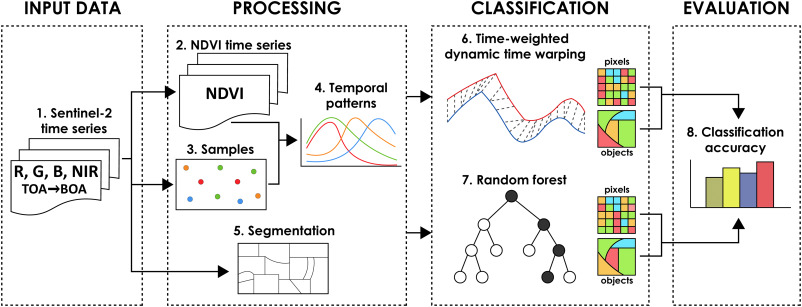
**Web Application Development:** In parallel with the remote sensing analysis, the project aims to develop a web application that offers proper analysis and visualisation of the selected forest area. The application will provide users with tools for interactive mapping, image processing, visualising forest change, and data visualisation. This objective aims to facilitate access to the analysed information, enhance decision-making processes, and support various people involved in forest management and conservation efforts.

# Project Overview

Our project aims to conduct a comprehensive analysis of forest dynamics using remote sensing data and machine learning techniques. The goal is to gain valuable insights into changes in forest area and understand the driving factors behind these changes. By leveraging the power of satellite imagery and advanced analytics, we aim to provide actionable information for effective forest management and conservation.

The project encompasses several key components. First, we acquire satellite imagery data from reliable sources, ensuring sufficient coverage over the study area. The data includes multispectral imagery capturing various spectral bands, enabling us to extract essential vegetation indices and environmental variables.

Next, we preprocess the data to remove noise, correct for atmospheric effects, and extract relevant features. Also, we calculate key vegetation indices such as NDVI, NDMI, and EVI, which serve as critical indicators of vegetation health, water content, and biomass.



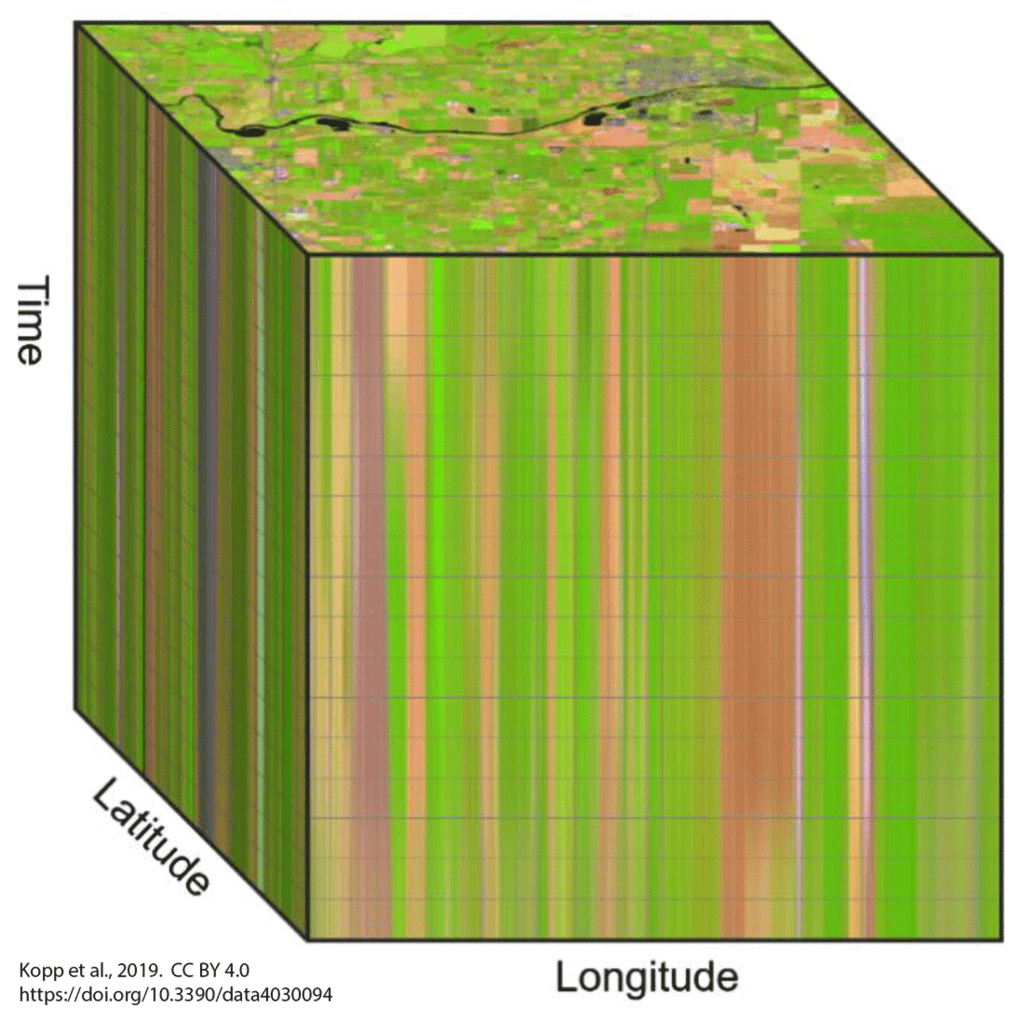
With the pre-processed data and derived features in hand, we employ machine learning algorithms, specifically the Random Forest algorithm, to model forest dynamics. By training the model on historical data and incorporating environmental variables, we can predict and analyse changes in forest area over time. The trained model provides valuable insights into the relative importance of different factors in driving forest dynamics.

Throughout the project, we emphasize the importance of accuracy, reliability, and interpretability. We employ rigorous validation techniques to assess the performance of our models and ensure the robustness of our results. Furthermore, we focus on clear and transparent communication of our findings, providing actionable information for forest managers, policymakers, and conservation organizations.

Our project relies on the data collected by various satellites. To efficiently store and access the massive amount of data, we have implemented a cutting-edge solution known as a Data Cube. The Data Cube serves as a powerful storage system specifically designed for time series analysis, ensuring seamless access and retrieval of data for our project's needs. We have used Flask in Python as our backend system in order to provide an API for our application to communicate with the Data Cube. We have also used Open Data Cube as our API interface in python to access data from the Data Cube.

# Introduction

## Data Cube

A data cube refers to a three-dimensional (3D) (or higher) range of values that are generally used to explain the time sequence of an image's data. It is a data abstraction to evaluate aggregated data from a variety of viewpoints. It is also useful for imaging spectroscopy as a spectrally resolved image is depicted as a 3-D volume. The Data cube pictorially shows how different attributes of data are arranged in the data model.

A data cube can also be described as the multidimensional extensions of two-dimensional tables. It can be viewed as a collection of identical 2-D tables stacked upon one another. Data cubes are used to represent data that is too complex to be described by a table of columns and rows. As such, data cubes can go far beyond 3-D to include many more dimensions.

Data cubes typically have dimensions with given properties common to different formats of spatial data. Typical properties include name, axis / number, type of data, extent/nominal dimension labels, reference system or projection, and resolution.

## Satellite Images and their Analysis

Satellite images play a vital role in remote sensing projects by providing a comprehensive and objective view of the Earth's surface from a considerable distance. Remote sensing involves the collection and analysis of information about the Earth's features, environment, and phenomena without direct physical contact.

Earth observatory satellites are a category of satellites specifically designed to observe and monitor the Earth's surface, atmosphere, and various environmental phenomena. The satellites we considered for our project are Landsat and sentinel-2.

### Sentinel-2

The Sentinel-2 mission, part of the European Space Agency's (ESA) Copernicus program, consists of two identical satellites, Sentinel-2A and Sentinel-2B. These satellites provide high-resolution multispectral imagery, enabling detailed land cover mapping, agricultural monitoring, and environmental management.

Sentinel-2 captures data in 13 spectral bands. These bands include the Coastal Aerosol band (sensitive to suspended matter in water), Blue band (for water and vegetation reflectance), Green band (vegetation health), Red band (vegetation analysis and soil moisture), and several Red Edge bands (precise vegetation health assessment).

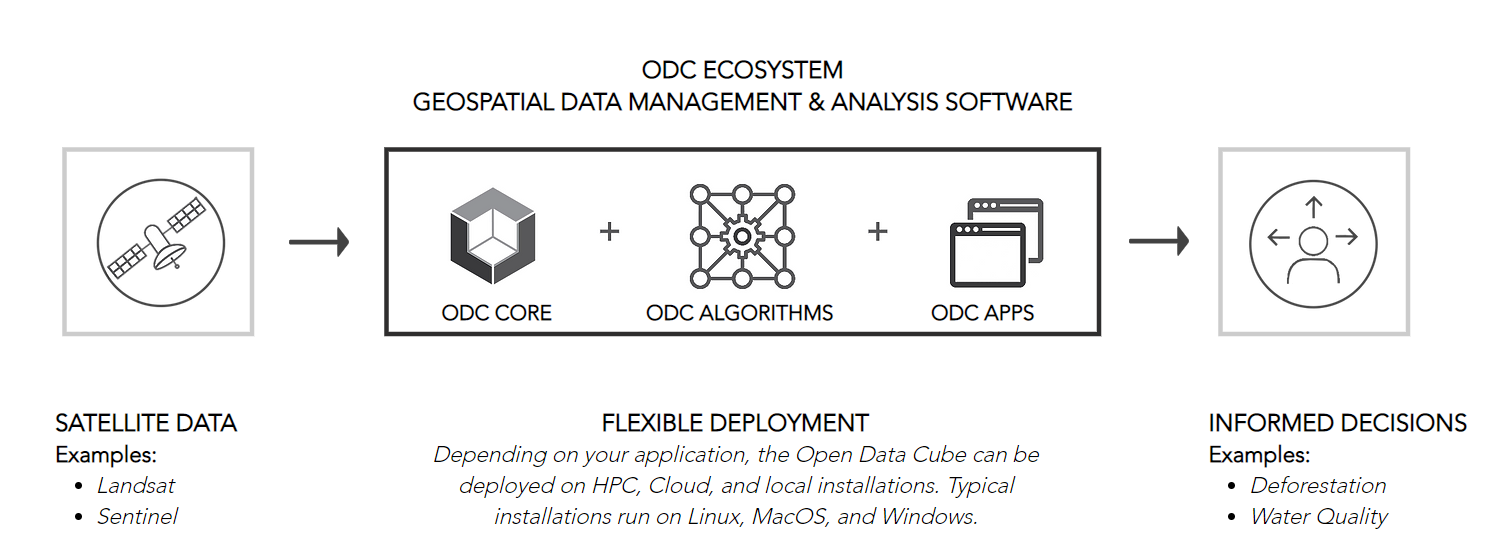
### Landsat

Landsat satellites, operated by the U.S. Geological Survey (USGS) and NASA, have been capturing imagery of the Earth's surface since 1972. The Landsat program provides a long-term record of the planet's changes, monitoring land cover, vegetation health, urban growth, and other land-related phenomena. Landsat 8 and Landsat 9 are the current active satellites in the series.

Landsat satellites also capture data in multiple spectral bands. These include visible, near-infrared, short-wave infrared, and thermal infrared bands. Each band represents specific characteristics such as land cover, vegetation health, water bodies, and thermal information.

## Open Data Cube

The Open Data Cube (ODC) is an open-source software package for organizing and analyzing large quantities of Earth observation data. At its core, the Open Data Cube consists of a database where data is stored, along with commands to load, view and analyse that data. This functionality is delivered by the data cube-core open-source Python library.



The ODC platforms are designed to:

* Catalogue large amounts of Earth observation data.
* Provide a Python based API for high performance querying and data access.
* Give users easy ability to perform exploratory data analysis.
* Allow scalable continent-scale processing of the stored data.
* Track the provenance of data to allow for quality control and updates.

## Time Series Analysis

Time series analysis refers to tools and methods to extract information about the landscape characterized by both spectral and temporal changes.

## Analysis Ready Data

The [ARD standard](http://ceos.org/ard/) for satellite data requires that data have undergone several processing steps, along with the creation of additional attributes for the data. ARD datasets include the following characteristics:

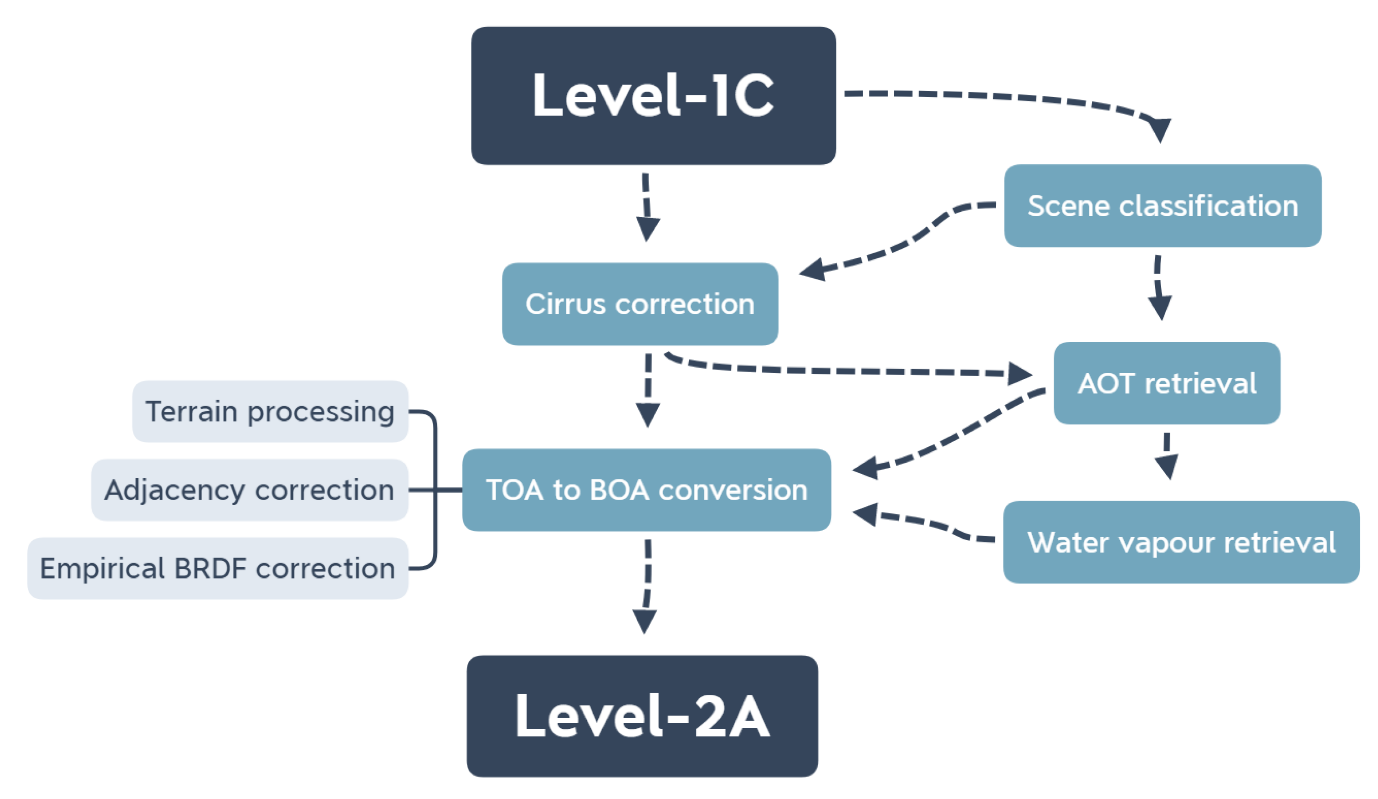
* **Geometric correction:** This includes establishing ground position, accounting for terrain (orthorectification) and ground control points and assessing absolute position accuracy. Geometric calibration means that imagery is positioned accurately on the Earth's surface and stacked consistently so that sequential observations can be used to track meaningful change over time. Adjustments for ground variability typically use a Digital Elevation Model (DEM).
* **Surface reflectance correction:** This includes adjustments for sensor/instrument gains, biases, and offsets, include adjustments for terrain illumination and sensor viewing angle with respect to the pixel position on the surface. Once satellite data is processed to surface reflectance, pixel values from the same sensor can be compared consistently both spatially and over time.
* **Observation attributes:** Per-pixel metadata such as quality flags and content attribution that enable users to make informed decisions about the suitability of the products for their use. For example, clouds, cloud shadows, missing data, saturation, and water are common pixel level attributes.
* **Metadata:** Dataset metadata including the satellite, instrument, acquisition date and time, spatial boundaries, pixel locations, mode, processing details, spectral or frequency response and grid projection.

## System Workflow

## Sen2Cor

The temporal resolution of a satellite in orbit is the revisit frequency of the satellite to a particular location. The revisit frequency of each single SENTINEL-2 satellite is 10 days and the combined constellation revisit is 5 days. Conversion from Sentinel 2 raw data into ARD product is done with Sen2Cor.

Satellites enable image acquisition over the same area every 5 days or less (Sentinel-2 has a revisit time of 5 days). To use the unique potential of the Sentinel-2 data for land applications and ensure the highest quality of scientific exploitation, accurate correction of satellite images for atmospheric effects is required. Therefore, the atmospheric correction processor Sen2Cor was developed by Telespazio VEGA Deutschland GmbH on behalf of ESA.



Sen2Cor is a Level-2A processor which main purpose is to correct single-date Sentinel-2 Level-1C Top-Of-Atmosphere (TOA) products from the effects of the atmosphere to deliver a Level-2A Bottom-Of-Atmosphere (BOA) reflectance product. Additional outputs are an Aerosol Optical Thickness (AOT) map, a Water Vapour (WV) map and a Scene Classification (SCL) map with Quality Indicators for cloud and snow probabilities. Its output product format is equivalent to the Level 1C User Product: JPEG 2000 images, three different resolutions, 60, 20 and 10 m.

# Setting up the Data Cube

Setting up the Data Cube involves many steps which includes setting up a Python environment and installing necessary modules, getting the required satellite images, adding them up into the Data Cube, visualising them in Data Cube Explorer in order to conform their installation and presence of datasets.

Data Cube can be set up in two ways. One by directly setting up in an Ubuntu machine. Other, by setting it up using Cube in a box, which is a Docker based container setup. We can choose either of them, which doesn’t matter for the Data Cube final product.

## Data Cube Setup in Ubuntu Operating System

**Datacube** runs the best on Ubuntu version 20.04. There is also a third-party tested software called as OSGeoLive VM which is based on Lubuntu 22.04. This provides pre-configured applications for a range of geo-spatial use-cases that includes storage, publishing, viewing, analysis and manipulation of data.

The steps that involve in setting up a Data Cube are:

1. Setting up the Python Environment
2. Postgres Database configuration for testing
3. Data Cube installation
4. Verification of the installation
5. Creating a Database for the actual Data Cube
6. Adding Product Definitions
7. Adding Satellite imagery to the file system
8. Transforming Metadata to YAML
9. Add datasets to the Data Cube
10. Verify everything works

## Data Cube Setup in Docker Container

Data Cube can also be setup by using the pre-built cube-in-a-dash Docker Container. This can be found in the reference by Open Data Cube GitHub <https://github.com/opendatacube/cube-in-a-box>.

1. Clone the git repo [cube-in-a-box](https://github.com/opendatacube/cube-in-a-box)
2. Build a local environment: docker-compose build
3. Start a local environment: docker-compose up
4. Set up your local Postgres database

# Analysis through QGIS

QGIS is a powerful open-source geographic information system (GIS) software that supports various remote sensing capabilities. By following the instructions below, you will be able to load satellite images, perform basic preprocessing, conduct analysis, and visualize the results using QGIS.

## Installing QGIS

If you haven't installed QGIS yet, visit the official QGIS website (https://qgis.org) and download the appropriate version for your operating system. Follow the installation instructions to set up QGIS on your computer.

## Load Satellite Images

Launch QGIS. In the menu bar, click on "Layer" and select "Add Layer" > "Add Raster Layer."

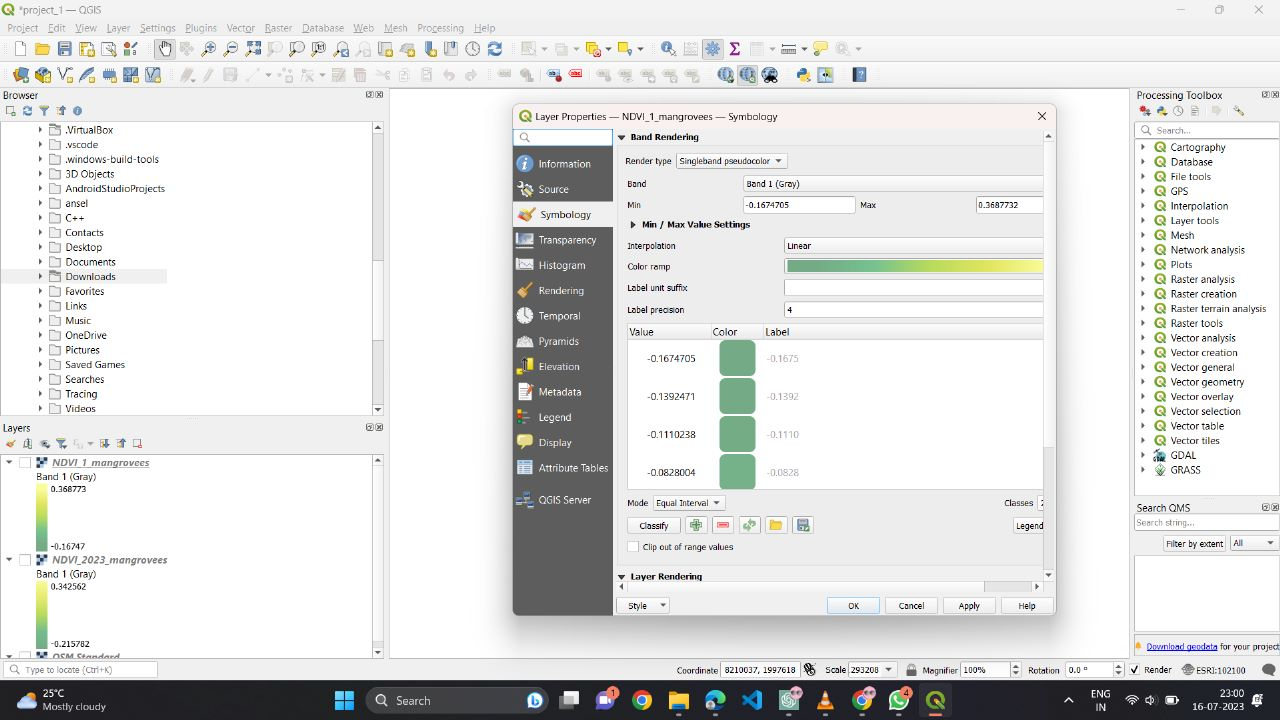
Browse and select the satellite image file you want to analyze. QGIS supports various satellite image formats, including GeoTIFF, JPEG, and PNG.

Click "Open" to load the satellite image into the QGIS canvas.

## Image Analysis

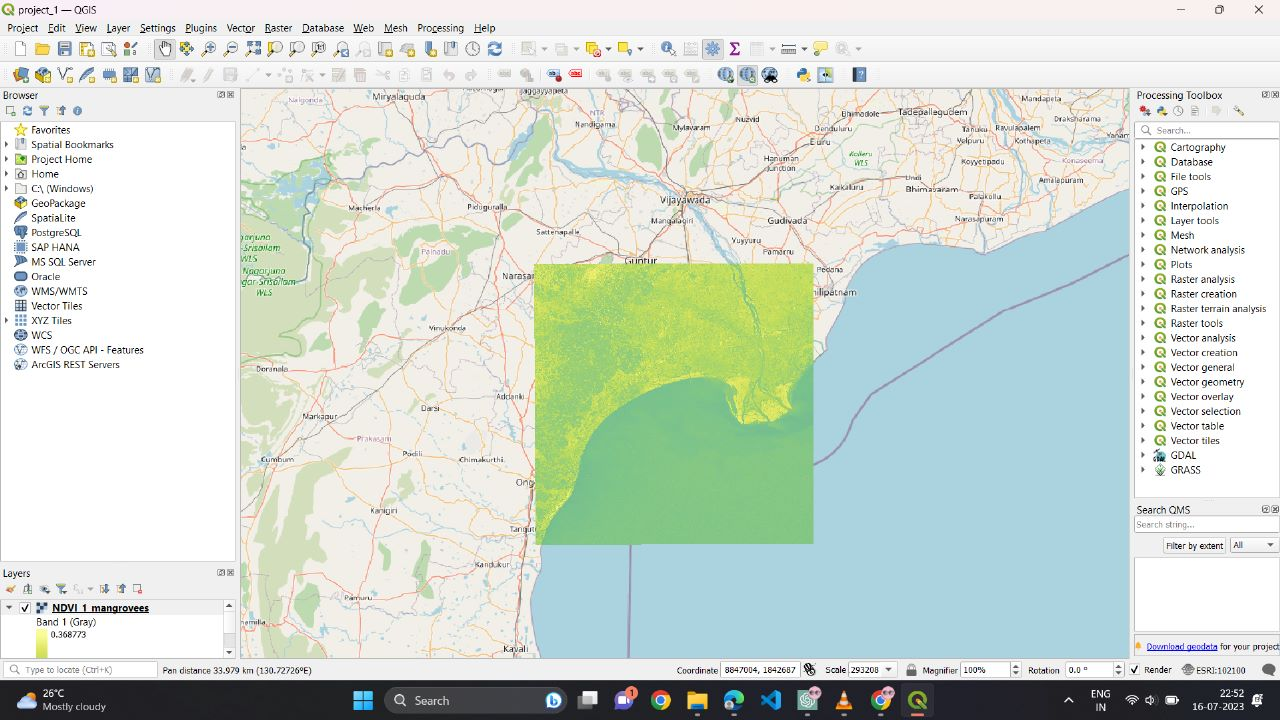
QGIS offers a wide range of tools and plugins for image analysis. Here are some commonly used tools and techniques:

1. Band Combinations: Create composite images by combining different bands to enhance certain features. Use the "Raster" > "Miscellaneous" > "Build Virtual Raster" tool to create a new composite image.
2. Image Classification: Perform image classification to categorise land cover types. QGIS provides plugins like Semi-Automatic Classification Plugin (SCP) or Orfeo Toolbox for image classification.
3. Indices Calculation: Calculate spectral indices like NDVI (Normalised Difference Vegetation Index) or NDBI (Normalised Difference Built-Up Index) to analyse vegetation or built-up areas. Use the "Raster Calculator" tool in the Processing Toolbox to perform these calculations.
4. Change Detection: Compare two or more satellite images to identify changes over time. The "Change Detection" plugin in QGIS helps in visualising and quantifying these changes.
5. Image Enhancement: Apply image enhancement techniques like contrast adjustment, histogram equalisation, or pansharpening to improve the visual quality of satellite images. These tools are available in the "Raster" menu.



## Visualization and Output

QGIS provides various visualization techniques to display and interpret the analysis results. Here are some visualization options:

* 1. Layer Styling: Adjust the symbology of the satellite image layer to visualise different bands or highlight specific features. Use the "Layer Styling" panel to modify colours, transparency, or blending modes.
  2. Add Additional Layers: Overlay vector data, such as shapefiles or point data, on top of the satellite image to provide context or perform spatial analysis.
  3. Generate Output: Export your analysis results or maps to different formats, such as PDF, image formats (PNG, JPEG), or geospatial formats (GeoTIFF, shapefile).

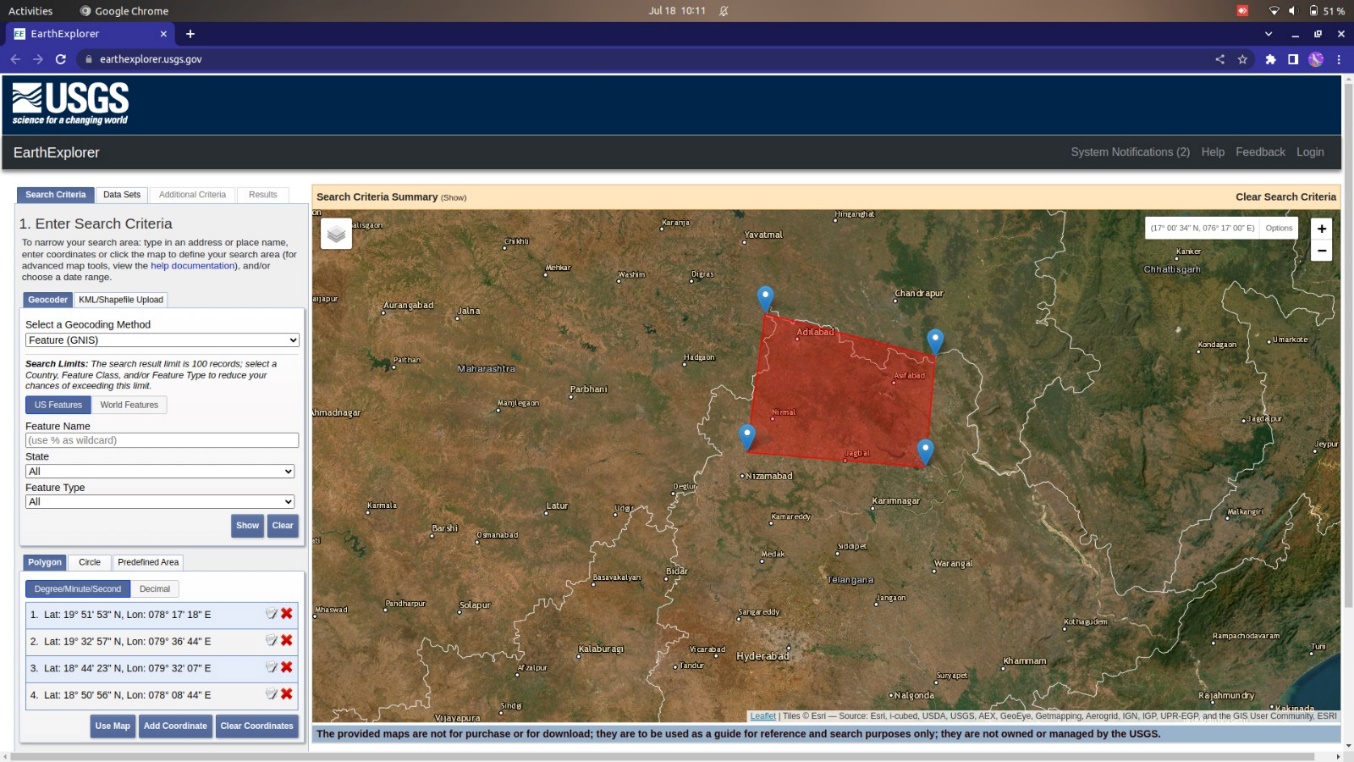
QGIS provides a comprehensive platform for analyzing satellite images and offers a wide range of functionalities to support remote sensing projects.

# Analysis using Manual Method

Analyzing satellite images can be done manually using Jupyter Lab, Data Cube, Xarray, and Matplotlib. By following the instructions below, we are able to extract satellite data from the Data Cube, process it using Xarray, and visualize vegetation changes using Matplotlib in a Jupyter Lab environment.

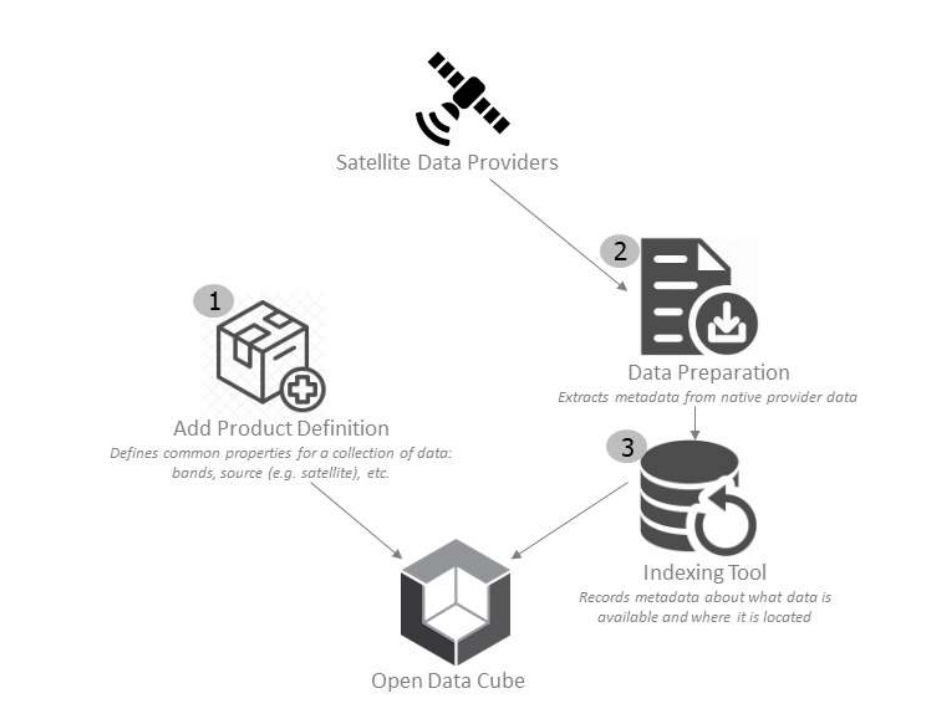
## Download and Index Satellite Data

Download the images from the respective websites of the satellites we need to add to the Data Cube we have already set up in our local system. The major satellite sources are USGS and ESA through their respective websites or through a command line script dhusget.sh.

* Landsat - <https://earthexplorer.usgs.gov/>
* Graphical user interface, application, website, map

  Description automatically generatedSentinel - <https://scihub.copernicus.eu/dhus>
* Command Line Script –

./dhusget.sh -d https://scihub.copernicus.eu/dhus -u rammohan503 -p Tele123$ -m Sentinel-2 -s 2019-01-05T00:00:000Z -e 2019-01-06T00:00:000Z -c 116.55,-21.83:116.80,-22.01 -l 100 -o product -O ./home/varsha/Desktop/

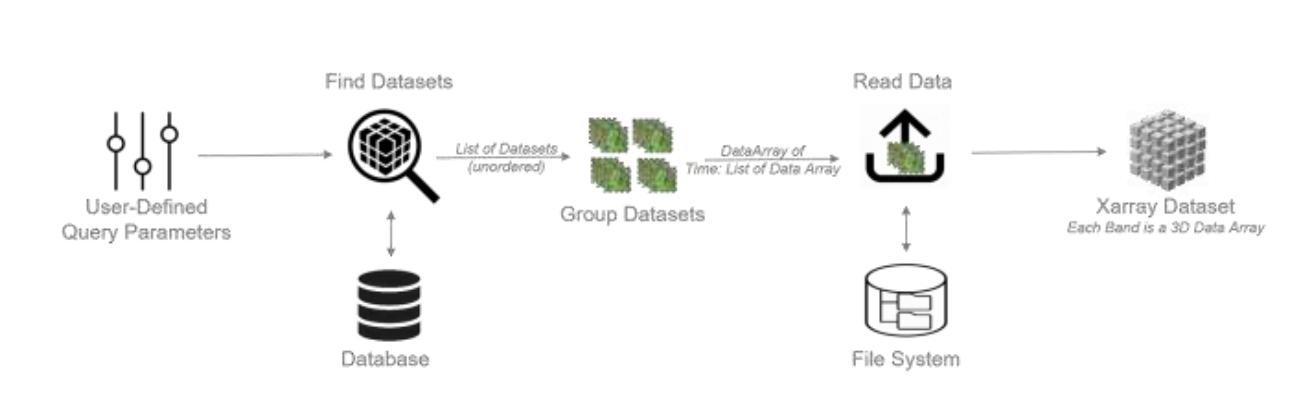


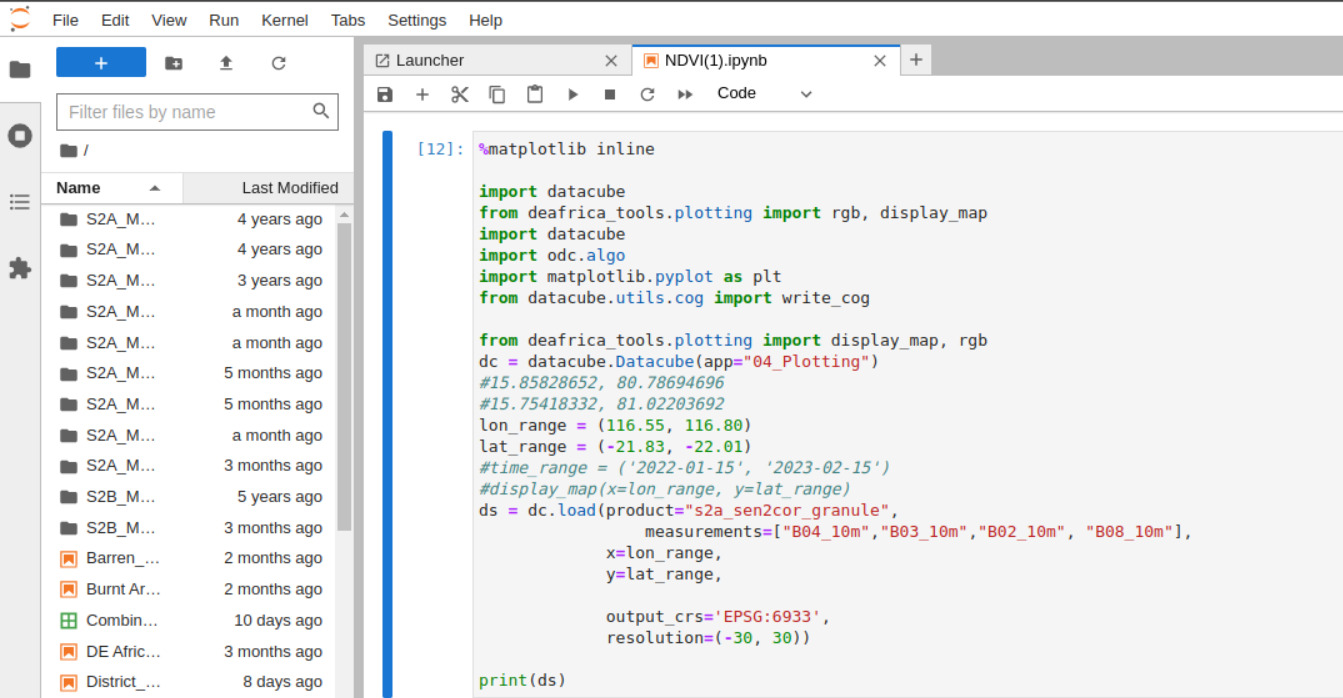
## Ingest the data locally

## Install Required Libraries

Ensure that you have Jupyter Lab, Data Cube, Xarray, and Matplotlib installed in your virtual environment. Import the necessary libraries in your Jupyter Notebook. Set up the Data Cube configuration to connect to your data source.

## Load Satellite Data

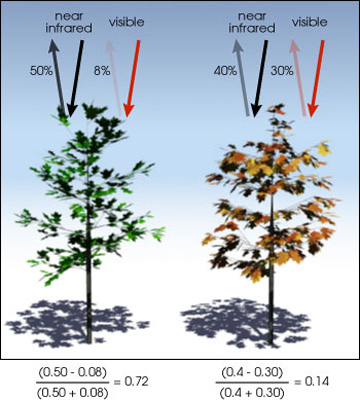
Specify the area of interest using coordinates or a bounding box and define the time range for analysis. Load the satellite data from the Data Cube using the APIs provided by the datacube module.



## Preprocess the Data

Preprocess the dataset as required. For vegetation analysis, you can calculate the Normalized Difference Vegetation Index (NDVI) using the data extracted from the Data Cube in the form of an Xarray.

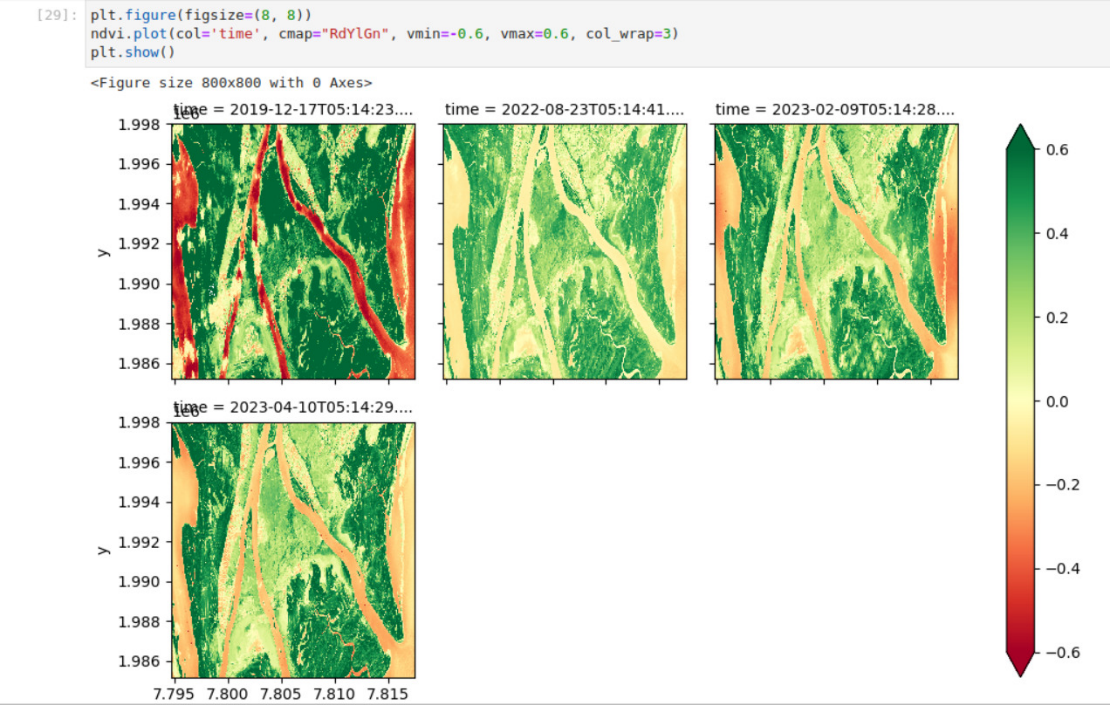
ndvi = (dataset["nir"] - dataset["red"]) / (dataset["nir"]+dataset["red"])



Perform additional preprocessing steps such as resampling, filtering, or masking, as per your analysis requirements.

## Visualize Results and Changes

Plot the vegetation change using Matplotlib image plot or datacube plot.



In this way, satellite data from the Data Cube can be extracted using the Data Cube APIs, processed it using Xarray functions, and vegetation changes can be visualized (for example) using Matplotlib in a Jupyter Lab environment. This workflow allows you to perform detailed satellite image analysis and gain insights into vegetation dynamics based on the specified coordinates area.

# Web Application

The web application involves in combination of several individual components such as the frontend, the backend and the Data Cube already set up using Postgres. This, hence, needs to be done in an efficient way such that we provide the best experience to the user and reducing the efforts and load on our systems.

## Frontend – User Interaction

The frontend part of the application is built by using the frontend technologies HTML, CSS, and JavaScript. Our requirements of the frontend frameworks are getting done by these three languages.

The User is expected to select an Area of Interest in the provided map. The User is then expected to enter the filter parameters such as start and end date for their query and type of analysis they wish to perform.

The map used in this application is integrated by using Leaflet JS library, which is being provided by Open Street Map. The application utilised the base map provided by the Open Street Map and then Leaflet JS allows us to add layers on top of the base map, in various formats of shapes, such as rectangle and polygon.

We also use Leaflet JS to indicate the bounding boxes of our available datasets in the Data Cube in order to mention user to select the Area of Interest from the indicated regions only. This helps us to maintain a decent User Experience.

The app connects with the backend through the APIs we define in the backend by sending asynchronous requests to them. The results are shown in the frontend using visualisation tools such as Plotly JS, Chart JS and Matplotlib plot.

## Backend – Flask API

The backend of the web application is built using Flask, a Python web framework. The Flask API handles incoming requests from the frontend, performs data processing and analysis, and provides responses accordingly. It interacts with the Data Cube with the provided APIs, which is already set up using Postgres, to retrieve relevant geospatial data based on the user's selections and filters.

The Flask API defines endpoints that are responsible for handling various requests from the frontend, such as retrieving available datasets, performing analysis based on the selected AOI and analysis type, and returning the results to the frontend. It utilizes various libraries and tools such as Data Cube, Postgres, scikit-learn, Matplotlib, and geopy for data processing, analysis, and visualization.

The Flask API ensures efficient handling of requests by optimizing the data retrieval and processing operations, thereby providing a smooth User Experience and reducing the load on the system.

# Results

The web application provides users with a range of analysis types and presents the results in a visually appealing and interactive manner. Through the application's user-friendly interface, users can explore and interpret the analysis outcomes effortlessly. Here are the key results and features of the web application:

## Analysis Results

1. NDVI Analysis:
   * The Normalized Difference Vegetation Index (NDVI) analysis enables users to assess vegetation health and density.
   * The application generates NDVI maps that visualize the distribution of vegetation in the selected area of interest.
   * The NDVI values range from -1 to 1, with higher values indicating healthier and denser vegetation.
2. NDWI Analysis:
   * The Normalized Difference Water Index (NDWI) analysis aids in identifying water bodies and assessing water content.
   * NDWI maps are generated, providing insights into the presence and distribution of water in the selected area.
   * The NDWI values range from -1 to 1, with higher values indicating a higher concentration of water.
3. EVI Analysis:
   * The Enhanced Vegetation Index (EVI) analysis offers an advanced approach to evaluate vegetation health and density.
   * EVI maps are generated, providing a more accurate representation of vegetation conditions, especially in areas with high levels of aerosol content.
   * The EVI values range from -1 to 1, with higher values indicating healthier and denser vegetation.
4. Random Forest Analysis:
   * The Random Forest analysis utilizes machine learning techniques to predict forest cover and changes over time.
   * The application performs regression analysis based on the selected area and time range to estimate forest cover and future trends.
   * Interactive charts and graphs are generated using Plotly to illustrate the predicted forest cover and provide insights into future forest dynamics.

## Visualizations

* The web application leverages interactive maps powered by Leaflet JS to display the results of the analysis.
* Users can navigate the maps, zoom in/out, and explore the analysis outcomes at different spatial scales.
* The application also includes interactive charts and graphs created with Plotly, allowing users to visualize and analyse data trends in an intuitive manner.

## Data Interpretation

* The analysis results provide valuable information on vegetation health, water content, and forest dynamics in the selected area.
* Users can interpret the analysis outcomes by observing patterns, trends, and spatial variations in the displayed maps and visualizations.
* It is important to consider the context and any specific domain knowledge when interpreting the results and drawing meaningful insights.

## User Experience

* The web application offers a seamless and user-friendly experience for interacting with the analysis results.
* Users can easily select the area of interest, specify analysis parameters, and visualize the outcomes with minimal effort.
* The intuitive interface, combined with interactive elements, enhances user engagement and understanding of the analysis results.

## Performance and Scalability

* The web application is designed to handle large datasets efficiently, ensuring optimal performance even with complex analysis tasks.
* Backend optimizations, such as enabling fast data retrieval and processing.
* The system architecture allows for scalability, accommodating future growth and accommodating increased user demand.

# Future Scope

The web application lays a strong foundation for further enhancements and expansion of its capabilities. Here are some potential areas of future development and improvements:

## Additional Analysis Types

* Expand the range of analysis types offered by the application to provide users with more insights into geospatial data.
* Integrate advanced analysis algorithms and techniques for specialized applications, such as land cover classification, change detection, or anomaly detection.
* Incorporate user feedback and domain-specific requirements to identify new analysis types that would benefit the target user base.

## Enhanced Visualization

* Continuously improve and refine the visualizations presented to users for a more informative and engaging experience.
* Explore advanced visualization techniques, such as 3D mapping, heatmaps, or time series animations, to convey complex geospatial information effectively.
* Integrate additional visualization libraries or tools to offer more customization options and flexibility to users in visualizing analysis results.

## User-Driven Customization

* Provide users with more control and customization options in selecting analysis parameters, visualizations, and data presentation.
* Allow users to save and share their analysis configurations and results for collaboration or future reference.
* Implement user feedback mechanisms to gather insights and suggestions for further improvements and customization options.

## Performance Optimization

* Continuously optimize the backend infrastructure and algorithms to handle larger datasets and complex analysis tasks more efficiently.
* Implement parallel processing techniques or distributed computing frameworks to speed up data processing and analysis.

## Integration with External Data Sources

* Enable integration with external geospatial data sources, such as satellite data providers or public APIs, to expand the range of available datasets for analysis.
* Incorporate real-time data feeds or sensor data for near-real-time analysis and monitoring of dynamic phenomena.
* Establish partnerships or collaborations to access specialized datasets that cater to specific user requirements or niche applications.

## User Authentication and Access Control

* Implement user authentication and access control mechanisms to ensure data security and privacy.
* Allow users to save and manage their analysis projects, datasets, and preferences securely.
* Provide role-based access control to facilitate collaboration among multiple users with varying levels of permissions and privileges.

# References

* <https://deafrica.africageoportal.com>
* <https://docs.digitalearthafrica.org/en/latest/sandbox/notebooks/Beginners_guide/02_Products_and_measurements.html>
* <https://sandbox.digitalearth.africa/hub/login?next=%2Fhub%2F>
* <https://www.dea.ga.gov.au/>
* <https://www.sentinel-hub.com/develop/custom-scripts/>
* <https://scihub.copernicus.eu/dhus/#/home>
* <https://www.opendatacube.org/>
* <https://datacube-core.readthedocs.io/en/latest/about-core-concepts/index.html>
* <https://opensourceoptions.com/blog/remote-sensing-with-qgis-calculate-ndvi/>
* <https://earthexplorer.usgs.gov/>
* <https://docs.dea.ga.gov.au/notebooks/Beginners_guide/08_Intro_to_xarray.html>
* <https://flask.palletsprojects.com/en/2.3.x/>