Assignment on Assignment Report on Mapping Crop-types using Available Satellite RS Data and VGI In-situ Data in R

Pranish Shrestha

2023-11-02

Executive Summary

In this report, We have conducted an analysis of mapping crop types using Sentinel satellite image data and volunteered geographic information (VGI) in R. We have used R, along with various packages, for geospatial analysis and machine learning to achieve the objectives. The primary objective was to classify different crop types based on the analysis using NDVI and in-situ observations.

Key Findings:

1. **Data Preparation**: We loaded in-situ data and have several proprocessing tasks such as cleaning, removing unusable column, and merging columns as required to represent land use and crop types.
2. **Data Visualization**: We used Leaflet to create an interactive map that displays the in-situ data points over OSM Map, categorized by land use and cover types with various color pallete to represent the data.
3. **Satellite Data**: We imported and processed Sentinel satellite data directly from web broswer and calculated the Normalized Difference Vegetation Index (NDVI).
4. **NDVI Analysis**: We calculated the NDVI values from the satellite data and created a heatmap to visualize vegetation health and land cover patterns.
5. **Machine Learning**: We implemented a random forest classifier model to predict land cover classes using NDVI as a predictor variable. The model was trained and tested using in-situ data in the ratio of 70%:30%.
6. **Model Evaluation**: We evaluated the model's performance using a confusion matrix and calculated accuracy, precision, recall, and F1-score for each land cover class. We also determined producer and user accuracy values.

Recommendations:

Based on the project findings, the following recommendations can be made:

- The more insitu data should be collect with more accuratly for refining the predictive model.
- Multiple indices should be used with addition of NDVI to have better predictions.
- Consider conducting field surveys and data collection to further validate the model's predictions and enhance its reliability.
- Extend the analysis to a larger geographical area to assess land use and land cover patterns more widely
- Explore with using more machine learning techniques to compare the results to have better reliability.

Table of Contents

Introduction

The effective management and monitoring of land use data is very crucial for urban planning as well as the ensuring the agricultural production with environment sustainability. With the accurate information regarding crop types and land use we chan have better decision-making in agriculture, resource allocation, and land planning. In this project,we have the utilized sentinel satellite remote sensing data and volunteered geographic information (VGI) to have Random Forest Classifier Model and predicted classsficiation of crop types and buildup areas

Context

This report focuses on a specific case study involving the mapping of crop types using available satellite remote sensing data and VGI data in the context of Madesh Province of Nepal specially Sarlahi, Dhanusha and Mohattari District. The in-situ data has been collected after the end of Mansoon in the month of October to Janurary. The sample size is approx 600, and is backed with photographic proof. The study area has been chosen due to its significance in agriculture and the presence of diverse crop types. The regions is known epicenter of for crop production for the country

Objectives

The primary objectives of this report are as follows:

- To analyze the feasibility and effectiveness of classifying crop types using satellite remote sensing data and VGI.
- To visualize the insitu data over the map
- To assess the accuracy and performance of the classification model with confusion matrix, overall accuracy and individual class performance.
- To analyse benifits and limitations of the approach and recommendations for further improvements in land use mapping.

Methods

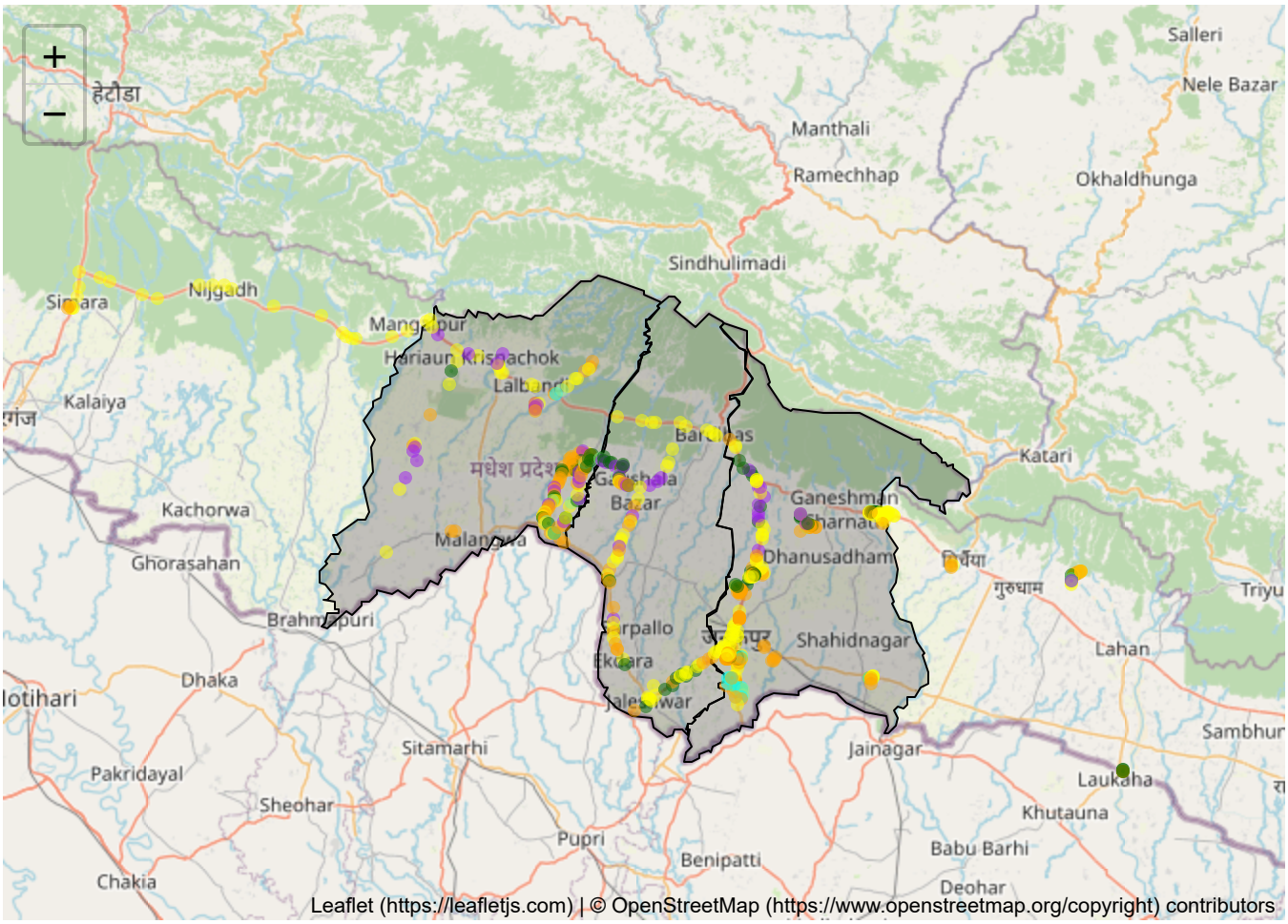
Data Collection:

Satellite Remote Sensing Data: The first component of our data collection involved obtaining satellite remote sensing data. We acquired multi-spectral Sentinel-2 imagery from https://dataspace.copernicus.eu/ (https://dataspace.copernicus.eu/) covering the study area during the month of December, 2021. Sentinel-2 is part of the European Space Agency's Copernicus program and provides high-resolution, multispectral imagery of the

Earth's surface. We selected an data of December, 2021 that aligns with our study objectives as our insitu data was on same time frame. The imagery consisted of various spectral bands, including red (B04) and near-infrared (B08), which are essential for calculating the normalized difference vegetation index (NDVI).

Volunteered Geographic Information (VGI): In parallel with satellite data, we had availed in-situ VGI data. The in-situ data used was collected after the end of the monsoon from October 23, 2021, to January 6, 2021. Participants provided valuable ground-level information about crop types in the study are with photograph and geographical coordinates associated with each observation.

Insitu Data Visualization



Analysis

In this section, we will describe the methods and techniques used to process and analyze the collected remote sensing and in-situ data. Our primary goal is to classify land use and crop types in the study area by integrating satellite imagery with in-situ data.

Libraries:

- **R Language:** We conducted our data analysis and geospatial processing using the R programming language. R is a powerful tool for statistical analysis and geospatial data handling, making it well-suited for our research objectives.
- **R Packages:** Several R packages were essential for our analysis.

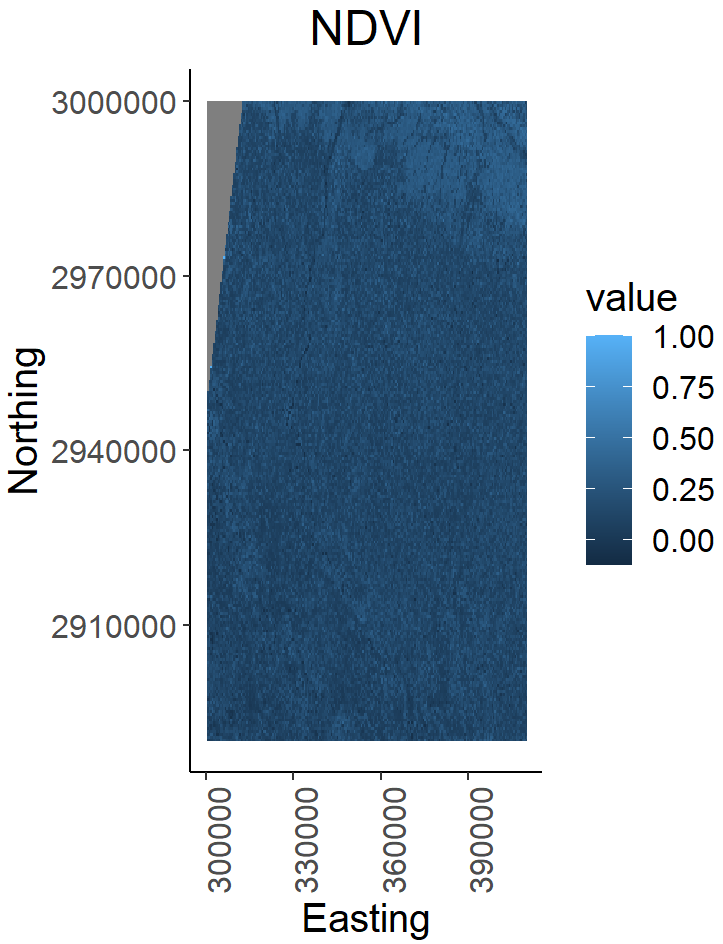
- **sf, terra and raster:** These packages allowed us to work with geospatial data, including shapefiles and raster imagery. We calculated the Normalized Difference Vegetation Index (NDVI) using these packages to assess vegetation health.
- **dplyr:** This package was used for data manipulation, allowing us to filter, mutate, and aggregate datasets effectively.
- **viridis:** We utilized the viridis package for defining custom color palettes in our visualizations, making it easier to distinguish between land cover classes.
- **randomForest:** We employed the random forest algorithm for supervised classification tasks.
- **knitr,tidyr,ggplot2:** These packages were used for data visualization, including the creation of plots, table and heatmaps.
- **leaflet,leaflet.extras:** For the interactive mapping component of our analysis, we utilized the leaflet package.

Data Preprocessing:

The data was clean, check for unusable columns and removed it. The new column LU_CT was created to store crop types and other build up area merges as Mixed_area.

Data Projection and Alignment: To ensure data compatibility, the satellite remote sensing data and VGI were projected and aligned to a common coordinate reference system (CRS, EPSG:32645).

NDVI Calculation: The NDVI was computed using the red and near-infrared bands of the satellite imagery. The NDVI calculation was performed using the formula (NIR - Red) / (NIR + Red), where NIR represents the near-infrared reflectance values and Red represents the red band reflectance values. This index is very helpful for distinguishing vegetation health and land use. ### NDVI Visualitzation



Data Splitting The dataset was divided into training and testing sets in a proportion of 70% for training and 30% for testing. At first the combined dataset of ndvi and insitu data was cleaned by removing all nans. The splitting was done such that that each land cover class was represented in both the training and testing datasets. A balance between model learning and evaluation was maintained by preserving a diverse representation of land cover classes in the test set and train set.

Classification:

We employed the Random Forest classification algorithm to categorize land use and crop types. We trained the classification model using the NDVI values derived from the satellite data and the reference data set created from VGI. The Random Forest algorithm is capable in classification tasks. Our independent variables included the NDVI values derived from the satellite imagery.

Results

Classification Results:

The primary objective of this study was to classify land use and crop types using the integrated remote sensing and in-situ data. The Random Forest classification model achieved the following results:

Overall Accuracy: The overall accuracy of the classification model was approximately 47.10%.

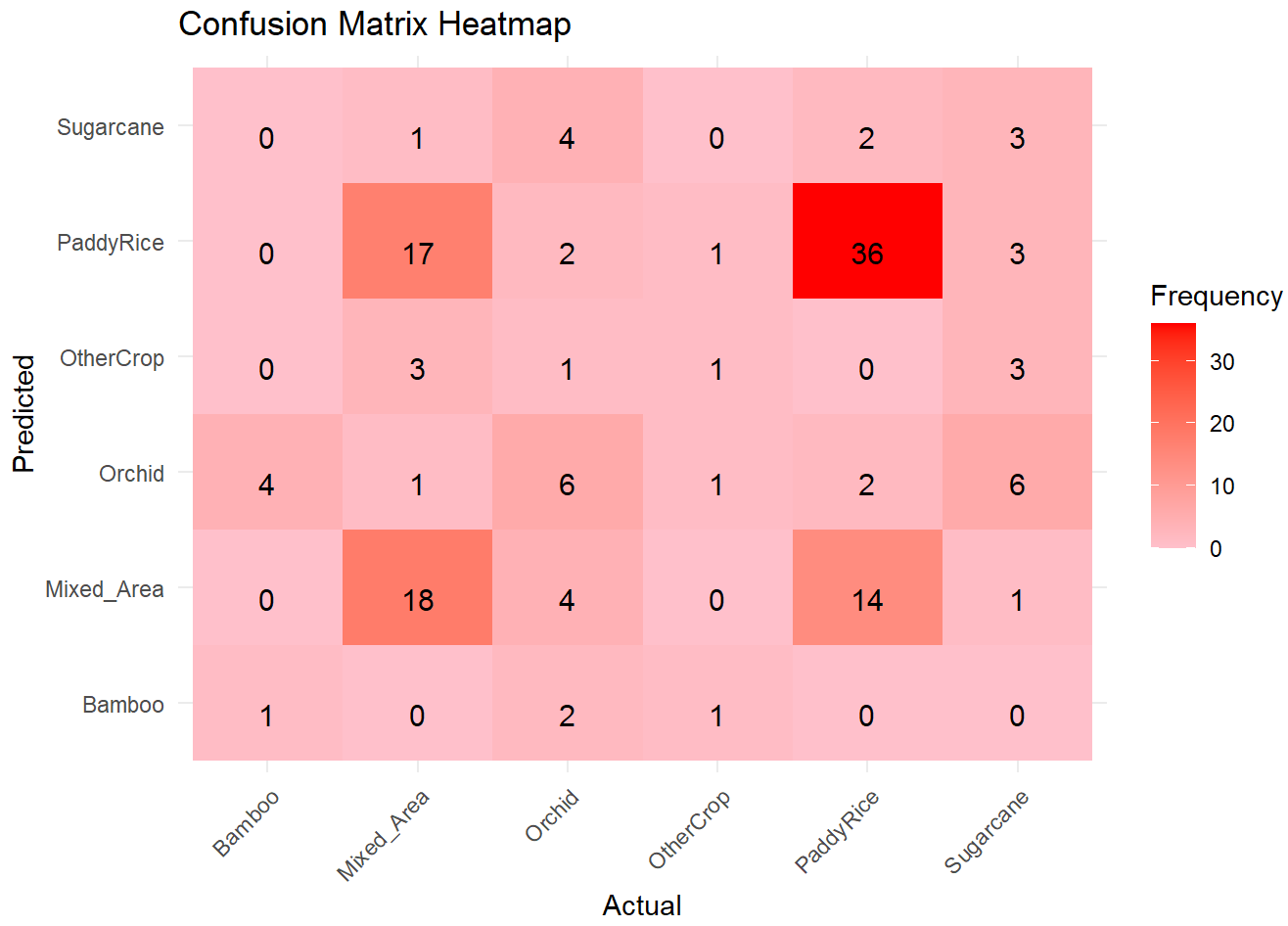
Confusion Matrix: The confusion matrix below provides detailed information on the model's performance. It displays the number of correctly and incorrectly classified samples for each class. It shows how well our model correctly classified different land types and helps us calculate important performance metrics like accuracy,

precision, recall, and F1-score. This matrix enables us to assess both the model's overall performance and its ability to correctly identify specific land use types.

```
##            Actual
## Predicted    Bamboo Mixed_Area Orchid OtherCrop PaddyRice Sugarcane
##   Bamboo          1          0      2         1         0         0
##   Mixed_Area      0         18      4         0        14         1
##   Orchid          4          1      6         1         2         6
##   OtherCrop       0          3      1         1         0         3
##   PaddyRice       0         17      2         1        36         3
##   Sugarcane       0          1      4         0         2         3
```

Heatmap Visualization of Confusion Matrix:

To evaluate the performance of our classification model, we created a heatmap of the confusion matrix. This heatmap visually represents the accuracy and misclassification of different land use and crop types.



Accuracy Assessment:

We assessed the accuracy of the classification model by calculating metrics such as overall accuracy, precision, recall, and F1-score for each land use and crop class. These metrics provide insights into the model's performance.

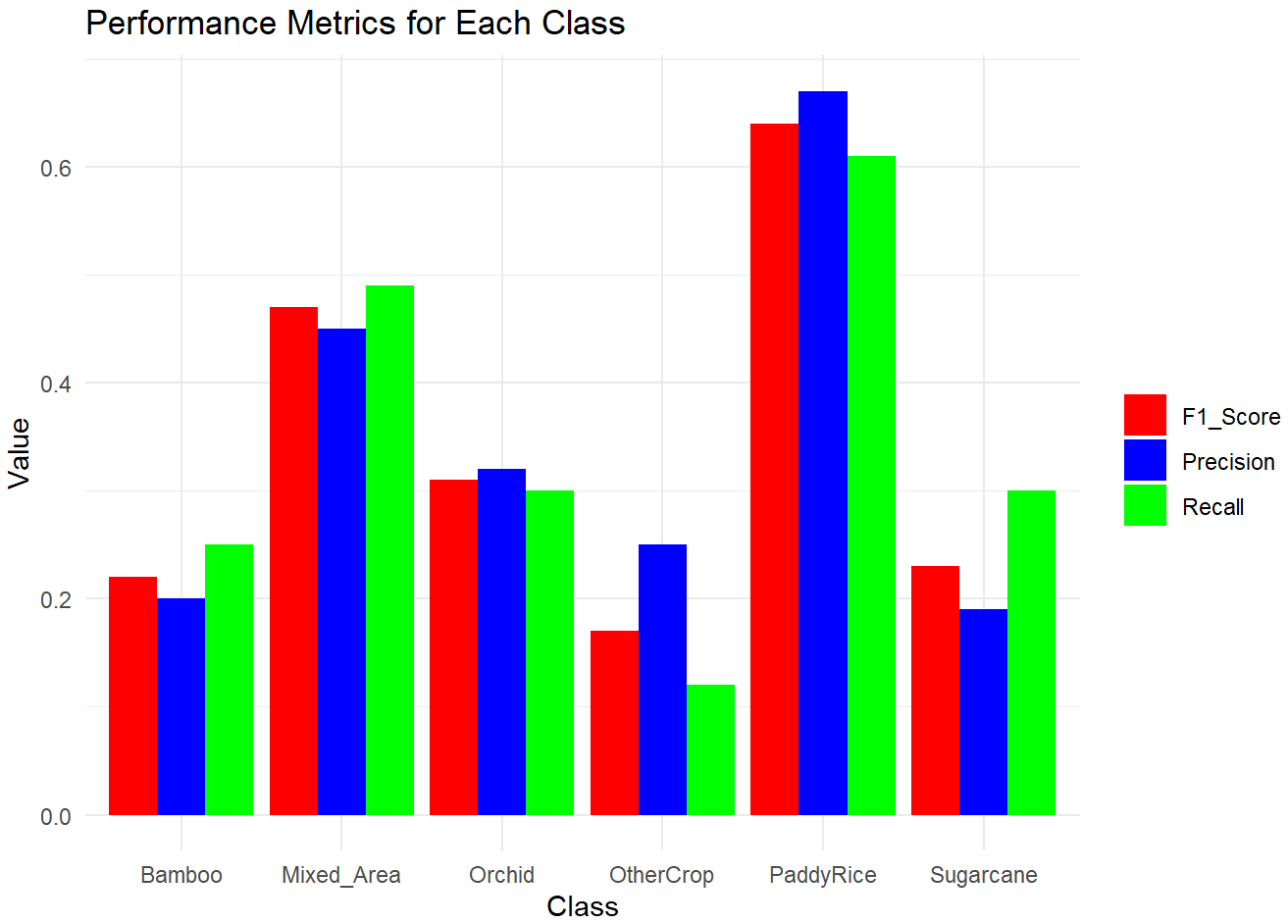
Precision, Recall, and F1-Score: Precision, recall, and F1-score were calculated for each land use and crop class. These metrics provide insights into the model's performance for individual classes:

- **Precision:** It measures the accuracy of our model's positive predictions. In our project, precision tells us how many of the predicted land use or land cover classifications are correct. A high precision score indicates a low rate of false positive predictions, which is crucial for accurate mapping.
- **Recall:** This metric assesses how well our model captures all the actual positive cases. In our case, it indicates how effectively our model identifies the various land use and land cover types present in the study area. High recall suggests that the model doesn't miss many true land cover categories.
- **F1-Score:** The F1-Score is the harmonic mean of precision and recall. It provides a balanced assessment of our model's performance, considering both false positives and false negatives. In our project, a high F1-Score is indicative of a balance model that accurately predicts land use and land cover categories without overly emphasizing precision or recall

Precision, Recall, and F1-Score for Each Class

Class	Precision	Recall	F1_Score
Bamboo	0.20	0.25	0.22
Orchid	0.32	0.30	0.31
PaddyRice	0.67	0.61	0.64
Sugarcane	0.19	0.30	0.23
OtherCrop	0.25	0.12	0.17
Mixed_Area	0.45	0.49	0.47

Bar Chart For Precision, recall and F1 Score

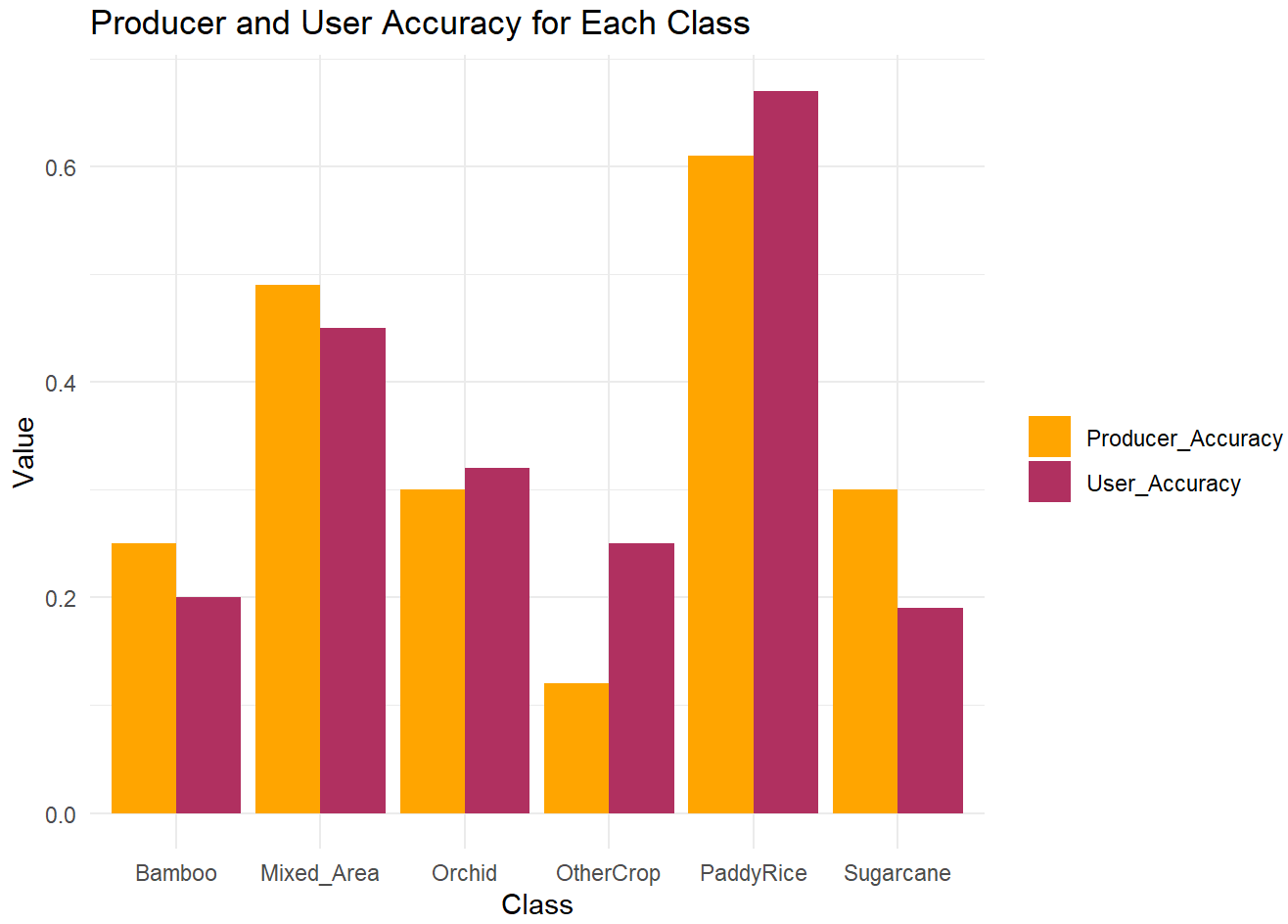


Producer Accuracy and User Accuracy: Producer Accuracy and User Accuracy were calculated for each land use and crop class. These metrics help assess the model's performance in terms of omission and commission errors which have varying implications for different applications. High UA values indicate that the model's predictions align well with actual land cover types, providing confidence in the model's outputs.These metrics are as follows.

Producer and User Accuracy for Each Class

Class	Producer_Accuracy	User_Accuracy
Bamboo	0.25	0.20
Orchid	0.30	0.32
PaddyRice	0.61	0.67
Sugarcane	0.30	0.19
OtherCrop	0.12	0.25
Mixed_Area	0.49	0.45

Bar Chart for Producer and User Accuracy:



Discussion

The Random Forest model exhibited an accuracy of approximately 0.47, indicating that it correctly classified nearly half of the land use and land cover categories. The classification model demonstrates variable performance across different land use and crop classes. For instance, the “PaddyRice” class exhibits relatively higher precision, recall, and F1-score compared to other classes, indicating better classification results.

The “Sugarcane” class has a lower precision and recall, suggesting challenges in distinguishing this class accurately from others.

The overall accuracy of approximately 47.10% indicates a moderate classification performance, considering the complexity of land use and crop classification.

The confusion matrix provides valuable insights into the model’s performance, highlighting areas of improvement for specific land use classes.

Conclusion

This project highlights the effectiveness of using Random Forest machine learning models for land use and land cover classification. It demonstrates that integrating in-situ data with satellite imagery can significantly enhance classification accuracy. The model exhibited niceresults and also identifies areas for future planning. By continuing to refine the model and incorporating more extensive datasets, we can further explore the potential of remote sensing and machine learning for applications in environmental and urban areas. This work lays the foundation for future research and applications in land use and land cover classification.

Crop classification requires a large amount of high-quality in-situ reference data which is very difficult and time-consuming to collect even in normal conditions and almost impossible to gather by the experts during a lockdown and transport bans. However, local youths can fulfill the demand when they are equipped with the appropriate tools. In this study, the local youths collected quality in-situ reference data to classify freely available high-resolution satellite imageries such as Sentinel 2 for delineating crop classes. Voluntarily collected data by youths can be an inexpensive and easily accessible source of reference data for crop classification, even in case of transport restrictions. The approach, however, requires in-depth training to the potential volunteers and near real-time feedback for improved quality in-situ data and capturing accompanied photographs to support the data verification.

References

1. Volunteered In-Situ Data for Agriculture Crop Mapping: A Case Study of Nepal By U. S. Panday , A. K. Pratihast , J. Aryal , R. Kayastha
2. Pratihast, A. K., DeVries, B., Avitabile, V., de Bruin, S., Kooistra, L., Tekle, M., & Herold, M. (2014). Combining satellite data and community-based observations for forest monitoring. Forests. https://doi.org/10.3390/f5102464 (https://doi.org/10.3390/f5102464)
3. Pratihast, A. K., Herold, M., Avitabile, V., de Bruin, S., Bartholomeus, H., Souza, C. M., & Ribbe, L. (2013). Mobile devices for community-based REDD+ monitoring: A case study for central Vietnam. Sensors (Switzerland), 13, 21–38. https://doi.org/10.3390/s130100021 (https://doi.org/10.3390/s130100021)

R code chunks:

Loading Required Libraries

```
# Load the required R packages
library(sf)      # For working with shapefiles
library(leaflet) # For geospatial mapping
library(rgdal)
library(raster)
library(terra)
library(dplyr)
library(ggplot2)
library(rgeos)
library(viridis)
library(rasterVis)
library(randomForest)
library(leaflet.extras)
library(knitr)
library(tidyr)
```

Loading Available insitu shp file and perform preprocessing task

Following tasks has been performed for preprocessing task in the insitu data

- Reviewed the first few rows of the attribute data.
- Listed all the column names in the shapefile.
- Removed non-usable columns from the data frame.
- Explored unique Crop Types and unique Landuse Types in the dataset.
- Created a new column, “NewColumn,” with a conditional statement that used “Agriculture” types from “data_CropT” and “data_LULC” values such that it stores crop types in place of agriculture.
- Created another column, “LU_CT,” based on a another conditional statement that categorized all other land use types into “Mixed_Area” but retaining the crop types values.
- Removed non-usable columns once again from the data frame.

```
#Load the in-situ data (shapefile)
insitu_shp <- st_read("LULC_Crop-Types_In-SituData2021_USP/LULC_Crop-Types_In-SituData2021_USP.shp")
```

```
## Reading layer `LULC_Crop-Types_In-SituData2021_USP' from data source
##   `D:\Geoinfomatics Study Materials\Second Semster\Geo_Science\RWorking\Pranish Final Project - LULC Classficiation\LULC_Crop-Types_In-SituData2021_USP\LULC_Crop-Types_In-SituData2021_USP.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 603 features and 14 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 84.96197 ymin: 26.44534 xmax: 86.94405 ymax: 27.19818
## Geodetic CRS:  WGS 84
```

```
# View the first few rows of the attribute data
head(insitu_shp)
```

```
## Simple feature collection with 6 features and 14 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 85.70018 ymin: 26.96397 xmax: 85.70945 ymax: 26.9662
## Geodetic CRS:  WGS 84
##   data_Start   data_LULC data_CropT data_Other
## 1 2021-10-23 Residential       <NA>       <NA>
## 2 2021-10-24 Agriculture  Sugarcane       <NA>
## 3 2021-10-24 Agriculture  PaddyRice       <NA>
## 4 2021-10-24 Agriculture  Sugarcane       <NA>
## 5 2021-10-24 Agriculture  PaddyRice       <NA>
## 6 2021-10-24 Agriculture  PaddyRice       <NA>
##                                                    data_Photo
## 1 https://drive.google.com/open?id=1Kfu62SUKkKlOOu7VYjDr79Lfc40RvWKR
## 2 https://drive.google.com/open?id=1lZWf4hZsGFsUb2VGCUuIqbii1QaVdYBQ
## 3 https://drive.google.com/open?id=1XOaaE41_-rLwNpn6ST2M_WSUlAtT0eA7
## 4 https://drive.google.com/open?id=145mOqX3UCTXuL9J_koo0XKEHxASlfw7R
## 5 https://drive.google.com/open?id=151KhmKcnI1ibogr2cOO-dZo_X5mrWN6W
## 6 https://drive.google.com/open?id=15t42Pn11pu2z5T3SeC_QBjHVD6g2ab4L
##        data_UserN data_Phone data_Subsc             data_Email data_EndTi
## 1 Nabin Kumar Sah 9864143694       <NA> nabinsah2057@gmail.com 2021-10-23
## 2 Nabin Kumar Sah 9864143694       <NA> nabinsah2057@gmail.com 2021-10-24
## 3 Nabin Kumar Sah 9864143694       <NA> nabinsah2057@gmail.com 2021-10-24
## 4 Nabin Kumar Sah 9864143694       <NA> nabinsah2057@gmail.com 2021-10-24
## 5 Nabin Kumar Sah 9864143694       <NA> nabinsah2057@gmail.com 2021-10-24
## 6 Nabin Kumar Sah 9864143694       <NA> nabinsah2057@gmail.com 2021-10-24
##                                 data_meta_  F17  F18 ClassType
## 1 uuid:405526a0-e204-4173-9d50-feef4b582c44 <NA> <NA>        NV
## 2 uuid:560772d0-11d9-4144-a404-2130bd287d5f <NA> <NA>        SC
## 3 uuid:ad966e8d-2ccc-4ccd-ba2a-e85d7865b9ba <NA> <NA>        PD
## 4 uuid:eed339ca-c69f-4739-aa09-8959863495a1 <NA> <NA>        SC
## 5 uuid:8f8531ff-4d8a-4cee-b26d-e5e5bd4e9898 <NA> <NA>        PD
## 6 uuid:dc04b49b-ec36-40de-9d2c-dd0ef6abd54d <NA> <NA>        PD
##                     geometry
## 1  POINT (85.70945 26.9662)
## 2 POINT (85.70624 26.96397)
## 3 POINT (85.70592 26.96418)
## 4 POINT (85.70403 26.96514)
## 5 POINT (85.70291 26.96515)
## 6 POINT (85.70018 26.96597)
```

```
# List all the column names in shapefile
column_names <- names(insitu_shp)
print(column_names)
```

```
##  [1] "data_Start" "data_LULC"  "data_CropT" "data_Other" "data_Photo"
##  [6] "data_UserN" "data_Phone" "data_Subsc" "data_Email" "data_EndTi"
## [11] "data_meta_" "F17"        "F18"        "ClassType"  "geometry"
```

```
# Remove the non-usable columns from  data frame
insitu_shp <- insitu_shp %>%
  select(-data_Other, -data_Photo, -data_UserN, -data_Phone, -data_Subsc, -data_Email, -data_End
Ti, -data_meta_, -F17, -F18)

# See the unique Crop Types
unique_data_CropT <- unique(insitu_shp$data_CropT)
print(unique_data_CropT)
```

```
## [1] NA          "Sugarcane" "PaddyRice" "Orchid"    "Bamboo"    "OtherCrop"
```

```
# See the unique Landuse  Types
unique_data_LULC <- unique(insitu_shp$data_LULC)
print(unique_data_LULC)
```

```
## [1] "Residential" "Agriculture" "WaterBodies" "PublicUse"   "CULARCH"
## [6] "Forest"      "Commercial"  "Industrial"
```

```
insitu_shp <- insitu_shp %>%
  mutate(NewColumn = ifelse(data_LULC == "Agriculture", data_CropT, data_LULC))

# View the first few rows of the updated data frame
head(insitu_shp)
```

```
## Simple feature collection with 6 features and 5 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 85.70018 ymin: 26.96397 xmax: 85.70945 ymax: 26.9662
## Geodetic CRS:  WGS 84
##   data_Start   data_LULC data_CropT ClassType                  geometry
## 1 2021-10-23 Residential       <NA>        NV  POINT (85.70945 26.9662)
## 2 2021-10-24 Agriculture  Sugarcane        SC POINT (85.70624 26.96397)
## 3 2021-10-24 Agriculture  PaddyRice        PD POINT (85.70592 26.96418)
## 4 2021-10-24 Agriculture  Sugarcane        SC POINT (85.70403 26.96514)
## 5 2021-10-24 Agriculture  PaddyRice        PD POINT (85.70291 26.96515)
## 6 2021-10-24 Agriculture  PaddyRice        PD POINT (85.70018 26.96597)
##     NewColumn
## 1 Residential
## 2   Sugarcane
## 3   PaddyRice
## 4   Sugarcane
## 5   PaddyRice
## 6   PaddyRice
```

```
insitu_shp <- insitu_shp %>%
  mutate(LU_CT = ifelse(data_LULC %in% c("Residential", "WaterBodies", "PublicUse", "CULARCH",
"Forest", "Commercial", "Industrial"), "Mixed_Area", data_LULC))

# View the first few rows of the updated data frame
head(insitu_shp)
```

```
## Simple feature collection with 6 features and 6 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 85.70018 ymin: 26.96397 xmax: 85.70945 ymax: 26.9662
## Geodetic CRS:  WGS 84
##   data_Start   data_LULC data_CropT ClassType                  geometry
## 1 2021-10-23 Residential       <NA>        NV  POINT (85.70945 26.9662)
## 2 2021-10-24 Agriculture  Sugarcane        SC POINT (85.70624 26.96397)
## 3 2021-10-24 Agriculture  PaddyRice        PD POINT (85.70592 26.96418)
## 4 2021-10-24 Agriculture  Sugarcane        SC POINT (85.70403 26.96514)
## 5 2021-10-24 Agriculture  PaddyRice        PD POINT (85.70291 26.96515)
## 6 2021-10-24 Agriculture  PaddyRice        PD POINT (85.70018 26.96597)
##     NewColumn       LU_CT
## 1 Residential  Mixed_Area
## 2   Sugarcane Agriculture
## 3   PaddyRice Agriculture
## 4   Sugarcane Agriculture
## 5   PaddyRice Agriculture
## 6   PaddyRice Agriculture
```

```
insitu_shp <- insitu_shp %>%
  mutate(LU_CT = ifelse(NewColumn %in% c("Residential", "WaterBodies", "PublicUse", "CULARCH",
"Forest", "Commercial", "Industrial"), "Mixed_Area", NewColumn))

# View the first few rows of the updated data frame
head(insitu_shp)
```

```
## Simple feature collection with 6 features and 6 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 85.70018 ymin: 26.96397 xmax: 85.70945 ymax: 26.9662
## Geodetic CRS:  WGS 84
##   data_Start   data_LULC data_CropT ClassType                  geometry
## 1 2021-10-23 Residential       <NA>        NV  POINT (85.70945 26.9662)
## 2 2021-10-24 Agriculture  Sugarcane        SC POINT (85.70624 26.96397)
## 3 2021-10-24 Agriculture  PaddyRice        PD POINT (85.70592 26.96418)
## 4 2021-10-24 Agriculture  Sugarcane        SC POINT (85.70403 26.96514)
## 5 2021-10-24 Agriculture  PaddyRice        PD POINT (85.70291 26.96515)
## 6 2021-10-24 Agriculture  PaddyRice        PD POINT (85.70018 26.96597)
##     NewColumn      LU_CT
## 1 Residential Mixed_Area
## 2   Sugarcane  Sugarcane
## 3   PaddyRice  PaddyRice
## 4   Sugarcane  Sugarcane
## 5   PaddyRice  PaddyRice
## 6   PaddyRice  PaddyRice
```

```
# Again Remove the non-usable columns from  data frame
insitu_shp <- insitu_shp %>%
  select(-ClassType, -data_CropT, -data_LULC, -NewColumn)
head(insitu_shp)
```

```
## Simple feature collection with 6 features and 2 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 85.70018 ymin: 26.96397 xmax: 85.70945 ymax: 26.9662
## Geodetic CRS:  WGS 84
##   data_Start      LU_CT                  geometry
## 1 2021-10-23 Mixed_Area  POINT (85.70945 26.9662)
## 2 2021-10-24  Sugarcane POINT (85.70624 26.96397)
## 3 2021-10-24  PaddyRice POINT (85.70592 26.96418)
## 4 2021-10-24  Sugarcane POINT (85.70403 26.96514)
## 5 2021-10-24  PaddyRice POINT (85.70291 26.96515)
## 6 2021-10-24  PaddyRice POINT (85.70018 26.96597)
```

Visualising Area of Availabe insitu data on OSM Map

Our area of interest is on Sarlai, Dhanusha and Mahottari District.

A color palette has been created and plotted the data on the OSM map. Following color is assigned to each of the crop types.

- Sugarcane : Light Green
- PaddyRice : Yellow
- Orchid : Dark Green
- Bamboo : Cyan
- OtherCrop : Orange

- Mixed_Area : Purple

```
# Load the district shapefile
district_shp <- st_read("nepal_data/hermes_NPL_new_wgs_2.shp")
```

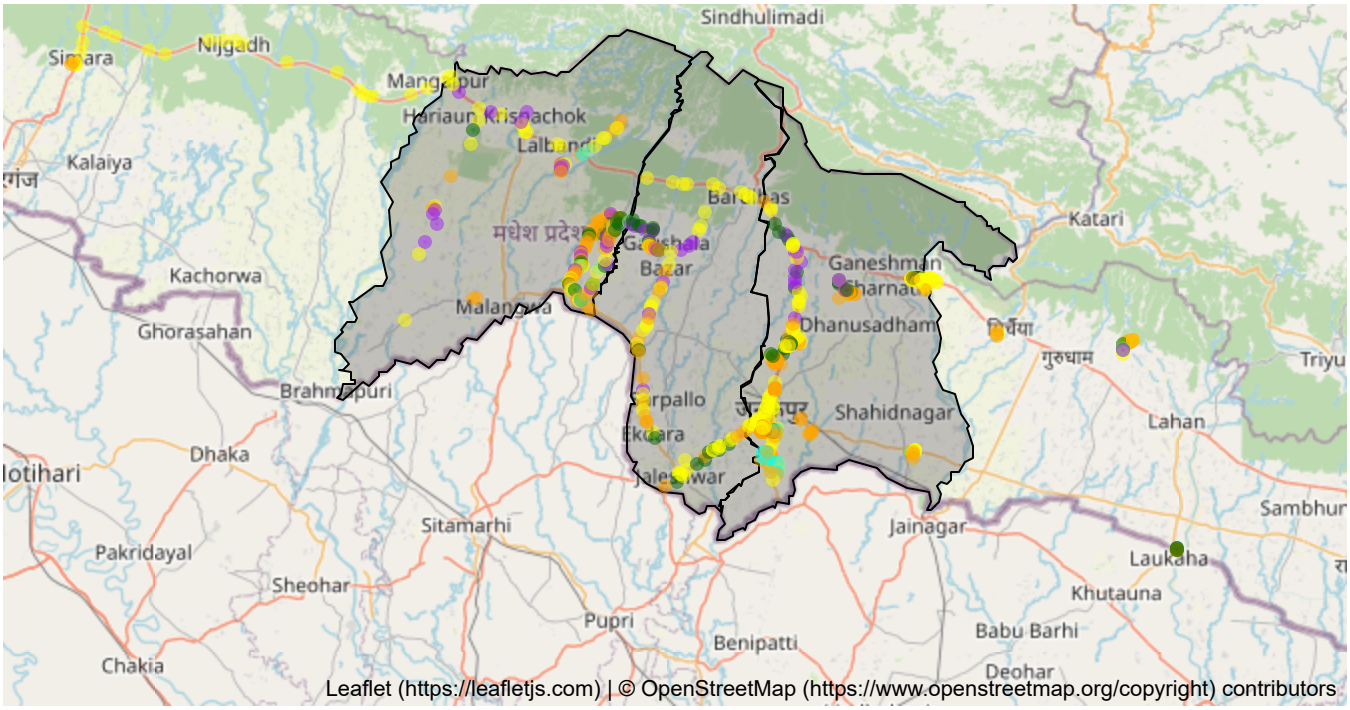
```
## Reading layer `hermes_NPL_new_wgs_2' from data source
##   `D:\Geoinfomatics Study Materials\Second Semster\Geo_Science\RWorking\Pranish Final Project
- LULC Classficiation\nepal_data\hermes_NPL_new_wgs_2.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 77 features and 4 fields
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: 80.06015 ymin: 26.34742 xmax: 88.2043 ymax: 30.46745
## Geodetic CRS:  WGS 84
```

```
# Define the target districts
target_districts <- c("Dhanusha", "Mahottari", "Sarlahi")

# Filter the shapefile to include only the target districts
filtered_districts <- district_shp[district_shp$DISTRICT %in% target_districts, ]

# Define a custom color palette for the specific class types
color_palette <- colorFactor(
  palette = c("lightgreen", "yellow", "darkgreen", "cyan", "orange","purple"),  # Define colors
for each class type
  domain = c("Sugarcane", "PaddyRice", "Orchid","Bamboo", "OtherCrop", "Mixed_Area")  # Define c
lass types
)

# Create a leaflet map with OpenStreetMap (OSM) as the base map
leaflet() %>%
  addProviderTiles(providers$OpenStreetMap) %>%
  addPolygons(data = filtered_districts, color = "black", weight = 1, opacity = 1, fillOpacity =
0.2) %>%
  addCircleMarkers(data = insitu_shp,
                   lng = ~st_coordinates(insitu_shp$geometry)[, 1],
                   lat = ~st_coordinates(insitu_shp$geometry)[, 2],
                   popup = ~LU_CT,
                   color = ~color_palette(LU_CT),
                   radius = 1)%>%
  setView(lng = 85.793, lat = 26.953, zoom = 8.8)
```



Satellite Data Collection and NDVI Calalculation

For this project Sentinel image has been downloaded. I have downloaded directly from the web browser and imported. Then NDVI is calculated and the NDVI image is with ggplot.

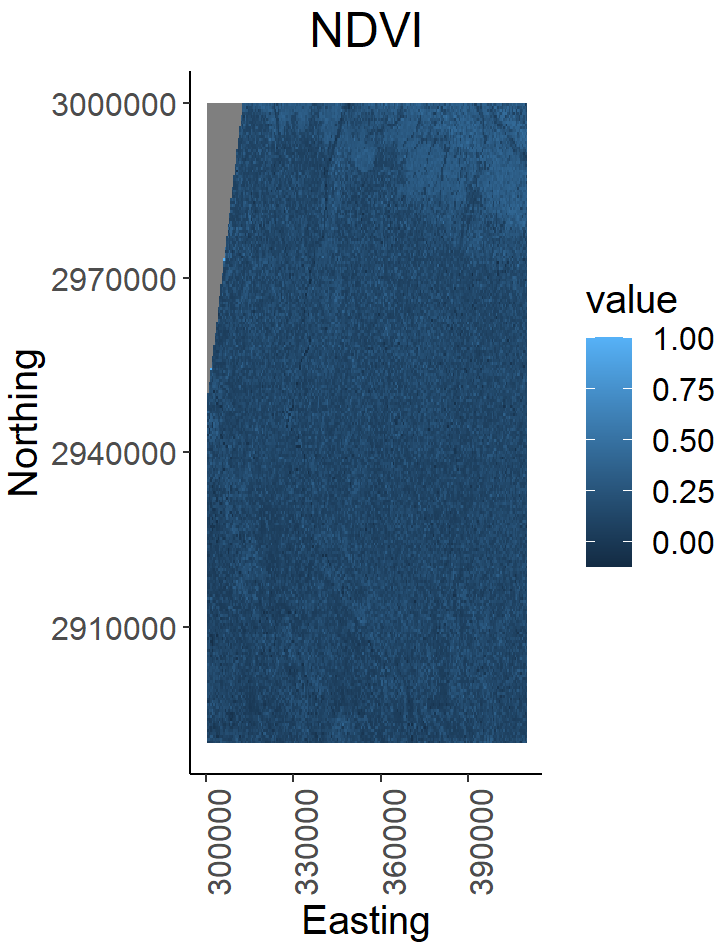
```
#importing sentinel image data

red_band <- raster("T45RUK_20211223T045221_B04.jp2")
nir_band <- raster("T45RUK_20211223T045221_B08.jp2")
TCI <- raster("T45RUK_20211223T045221_TCI.jp2")

# Calculation NDVI
ndvi_funtion <- function(red_band, nir_band) {
  ndvi <- (nir_band - red_band) / (nir_band + red_band)
  return(ndvi)
}

ndvi <- ndvi_funtion(red_band , nir_band)

#plotting NDVI
gplot(ndvi) +
  geom_raster(aes(x = x, y = y, fill = value)) +
  coord_quickmap() +
  ggtitle("NDVI") +
  xlab("Easting") +
  ylab("Northing") +
  theme_classic() +
  theme(plot.title = element_text(hjust = 0.5),
        text = element_text(size=15),
        axis.text.x = element_text(angle = 90, hjust = 1))
```



Data processing with spatial alignment

On this step following task has been done

- Checked and confirmed the Coordinate Reference System (CRS) of the NDVI data.
- Checked and confirmed the CRS of the in-situ shapefile.
- Defined the target CRS, which is EPSG:32645.
- Projected the in-situ data to the target CRS using the st_transform function.
- Extracted NDVI values for the locations specified in the projected in-situ data.
- Created a new data frame named insitu_combined by combining the projected in-situ data and the extracted NDVI values.
- Checked the first few rows of the insitu_combined data frame.
- Removed rows with NA (Not Available) values in the NDVI column so that the area beyond the used satellite image can be avoided.
- Checked the number of rows in the insitu_combined data frame after removing NA values.

```
# Check the CRS ofNDVI data
crs_ndvi <- st_crs(ndvi)
# Print the CRS
print(crs_ndvi)
```

```
## Coordinate Reference System:
##   User input: +proj=utm +zone=45 +datum=WGS84 +units=m +no_defs
##   wkt:
## PROJCRS["unknown",
##     BASEGEOGCRS["unknown",
##         DATUM["World Geodetic System 1984",
##             ELLIPSOID["WGS 84",6378137,298.257223563,
##                 LENGTHUNIT["metre",1]],
##             ID["EPSG",6326]],
##         PRIMEM["Greenwich",0,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8901]]],
##     CONVERSION["UTM zone 45N",
##         METHOD["Transverse Mercator",
##             ID["EPSG",9807]],
##         PARAMETER["Latitude of natural origin",0,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8801]],
##         PARAMETER["Longitude of natural origin",87,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8802]],
##         PARAMETER["Scale factor at natural origin",0.9996,
##             SCALEUNIT["unity",1],
##             ID["EPSG",8805]],
##         PARAMETER["False easting",500000,
##             LENGTHUNIT["metre",1],
##             ID["EPSG",8806]],
##         PARAMETER["False northing",0,
##             LENGTHUNIT["metre",1],
##             ID["EPSG",8807]],
##         ID["EPSG",16045]],
##     CS[Cartesian,2],
##         AXIS["(E)",east,
##             ORDER[1],
##             LENGTHUNIT["metre",1,
##                 ID["EPSG",9001]]],
##         AXIS["(N)",north,
##             ORDER[2],
##             LENGTHUNIT["metre",1,
##                 ID["EPSG",9001]]]]
```

```
# Check the CRS of shapefile
crs_insitu <- st_crs(insitu_shp)
# Print the CRS
print(crs_insitu)
```

```
## Coordinate Reference System:
##   User input: WGS 84
##   wkt:
## GEOGCRS["WGS 84",
##     DATUM["World Geodetic System 1984",
##         ELLIPSOID["WGS 84",6378137,298.257223563,
##             LENGTHUNIT["metre",1]]],
##     PRIMEM["Greenwich",0,
##         ANGLEUNIT["degree",0.0174532925199433]],
##     CS[ellipsoidal,2],
##         AXIS["latitude",north,
##             ORDER[1],
##             ANGLEUNIT["degree",0.0174532925199433]],
##         AXIS["longitude",east,
##             ORDER[2],
##             ANGLEUNIT["degree",0.0174532925199433]],
##     ID["EPSG",4326]]
```

```
crs_target <- st_crs("+init=EPSG:32645")

insitu_projected <- st_transform(insitu_shp, crs_target)


# Extract NDVI values for insitu data locations
ndvi_values <- raster::extract(ndvi, insitu_projected)

# Add the extracted NDVI values to the insitu_filtered_projected data
insitu_combined <- cbind(insitu_projected, NDVI = ndvi_values)

head(insitu_combined)
```

```
## Simple feature collection with 6 features and 3 fields
## Geometry type: POINT
## Dimension:     XY
## Bounding box:  xmin: 370992.9 ymin: 2983102 xmax: 371913.5 ymax: 2983345
## Projected CRS: WGS 84 / UTM zone 45N
##   data_Start      LU_CT      NDVI                 geometry
## 1 2021-10-23 Mixed_Area 0.0835746 POINT (371913.5 2983345)
## 2 2021-10-24  Sugarcane 0.2466019 POINT (371591.9 2983102)
## 3 2021-10-24  PaddyRice 0.1906907 POINT (371560.8 2983126)
## 4 2021-10-24  Sugarcane 0.2923643 POINT (371373.7 2983234)
## 5 2021-10-24  PaddyRice 0.1211256 POINT (371262.6 2983236)
## 6 2021-10-24  PaddyRice 0.1235089 POINT (370992.9 2983330)
```

```
nrow(insitu_combined)
```

```
## [1] 603
```

```
# Remove rows with NA values in the NDVI column
insitu_combined <- insitu_combined[!is.na(insitu_combined$NDVI), ]
nrow(insitu_combined)
```

```
## [1] 460
```

Data Splitting and Random Forest Model Training

- Defined the proportion for the training data (70% for training, 30% for testing).
- Set a seed for reproducibility.
- Initialized empty data frames for training and testing sets.
- Defined class names: “Bamboo,” “Orchid,” “PaddyRice,” “Sugarcane,” “OtherCrop,” and “Mixed_Area.”
- Splitted the data into training and testing sets with at least one sample from each class.
- For each class:
    - Filtered rows for the current class.
    - Calculated the number of samples for training.
    - Randomly selected samples for training and the rest for testing.
    - Add the selected samples to the training and testing sets.
    - Check the composition of the training and testing sets.
- Trained a random forest model. Converted the “LU_CT” variable to a factor.
- Trained the random forest model with “LU_CT” predicted based on the “NDVI” variable.I created the model with 100 trees (ntree = 100).

```
# Define the proportion for the training data (70% for training, 30% for testing)
train_prop <- 0.7

# Set a seed for reproducibility
set.seed(123)

# Initialize empty data frames for training and testing sets
train_data <- data.frame()
test_data <- data.frame()

# Define class names
class_names <- c("Bamboo", "Orchid", "PaddyRice", "Sugarcane", "OtherCrop", "Mixed_Area")

# Split the data into training and testing sets with at least one sample from each class
for (class_name in class_names) {
  # Filter rows for the current class
  class_data <- insitu_combined[insitu_combined$LU_CT == class_name, ]

  # Calculate the number of samples for training
  num_train_samples <- round(train_prop * nrow(class_data))

  # Randomly select samples for training and the rest for testing
  class_train_index <- sample(1:nrow(class_data), num_train_samples)
  class_test_index <- setdiff(1:nrow(class_data), class_train_index)

  # Add the selected samples to the training and testing sets
  train_data <- rbind(train_data, class_data[class_train_index, ])
  test_data <- rbind(test_data, class_data[class_test_index, ])
}

# Check the composition of the training and testing sets
table(train_data$LU_CT)
```

```
##
##     Bamboo Mixed_Area     Orchid  OtherCrop  PaddyRice  Sugarcane
##         11         94         46         10        125         36
```

```
table(test_data$LU_CT)
```

```
##
##     Bamboo Mixed_Area     Orchid  OtherCrop  PaddyRice  Sugarcane
##          5         40         19          4         54         16
```

```
# Train the random forest model
train_data$LU_CT <- as.factor(train_data$LU_CT)
rf_model <- randomForest(LU_CT ~ NDVI, data = train_data, ntree = 100)
```

###Model Prediction and Evaluation

- Made predictions on the test data using the trained random forest model.
- Created a confusion matrix to compare predicted and actual classes.
- Printed the confusion matrix as well as the head map.

```
# Make predictions on the test data
rf_predictions <- predict(rf_model, test_data)

predicted_classes <- predict(rf_model, test_data)
actual_classes <- test_data$LU_CT
confusion_matrix <- table(Predicted = predicted_classes, Actual = actual_classes)
print(confusion_matrix)
```

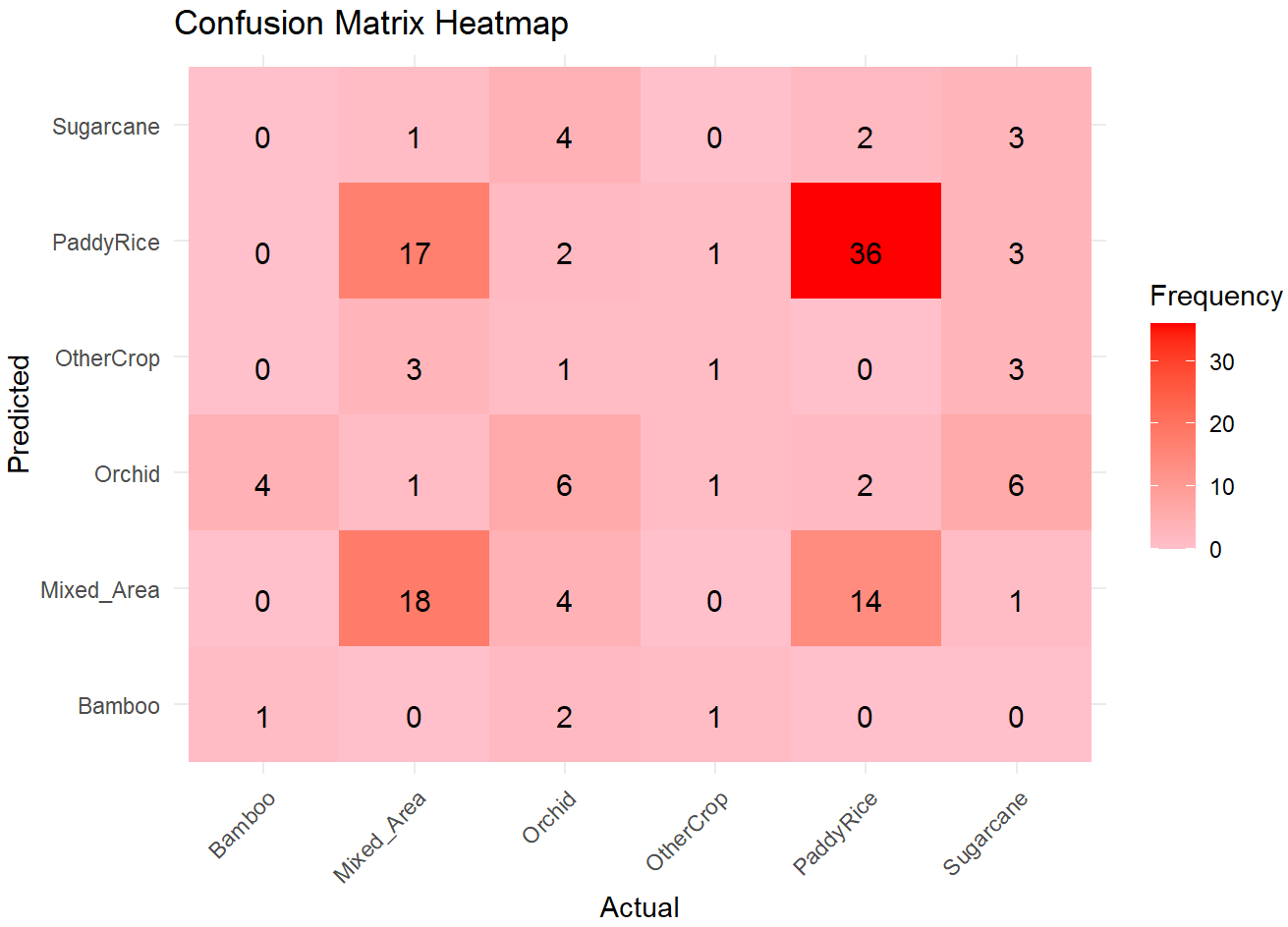
```
##             Actual
## Predicted    Bamboo Mixed_Area Orchid OtherCrop PaddyRice Sugarcane
##   Bamboo          1          0      2         1         0         0
##   Mixed_Area      0         18      4         0        14         1
##   Orchid          4          1      6         1         2         6
##   OtherCrop       0          3      1         1         0         3
##   PaddyRice       0         17      2         1        36         3
##   Sugarcane       0          1      4         0         2         3
```

```
# Convert the confusion matrix to a data frame
confusion_matrix_df <- as.data.frame(as.table(confusion_matrix))

# Rename the columns for better labels
colnames(confusion_matrix_df) <- c("Predicted", "Actual", "Frequency")

# Create the heatmap using ggplot2
heatmap_plot <- ggplot(confusion_matrix_df, aes(Actual, Predicted, fill = Frequency)) +
  geom_tile() +
  geom_text(aes(label = Frequency), vjust = 1, size = 4) +
  scale_fill_gradient(low = "pink", high = "red") +  # Define color palette
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +  # Rotate x-axis labels
  labs(title = "Confusion Matrix Heatmap", x = "Actual", y = "Predicted")

# Display the heatmap
print(heatmap_plot)
```



Calculation of Accuarcy, Precision, Recall, and F1-Score

- Calculated the overall accuracy of the model using the confusion matrix and displayed it to two decimal places.
- Calculated precision, recall, and F1-score values for each class using the confusion matrix and Stored these values in their respective vectors.
- Created a data frame to organize the precision, recall, and F1-score results for each class.
- Used the kable function to format the results as a table.
- Displayed the precision, recall and F1-Score value in form of Bar Diagram

```
# Calculate accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("Accuracy: ", round(accuracy, 2), "\n")
```

```
## Accuracy:  0.47
```

```
# Initialize vectors to store precision, recall, and F1 values
precision_values <- numeric(length(class_names))
recall_values <- numeric(length(class_names))
f1_values <- numeric(length(class_names))


for (i in 1:length(class_names)) {
  class_name <- class_names[i]
  TP <- confusion_matrix[class_name, class_name]
  FP <- sum(confusion_matrix[, class_name]) - TP
  FN <- sum(confusion_matrix[class_name, ]) - TP
  precision <- TP / (TP + FP)
  recall <- TP / (TP + FN)
  f1_value <- 2 * (precision * recall) / (precision + recall)



  # Store values in respective vectors
  precision_values[i] <- precision
  recall_values[i] <- recall
  f1_values[i] <- f1_value
}

# Create a data frame with  results
results_df <- data.frame(
  Class = class_names,
  Precision = round(precision_values, 2),
  Recall = round(recall_values, 2),
  F1_Score = round(f1_values, 2)
)

# Use kable to format the results as a table
kable(results_df, caption = "Precision, Recall, and F1-Score for Each Class")
```

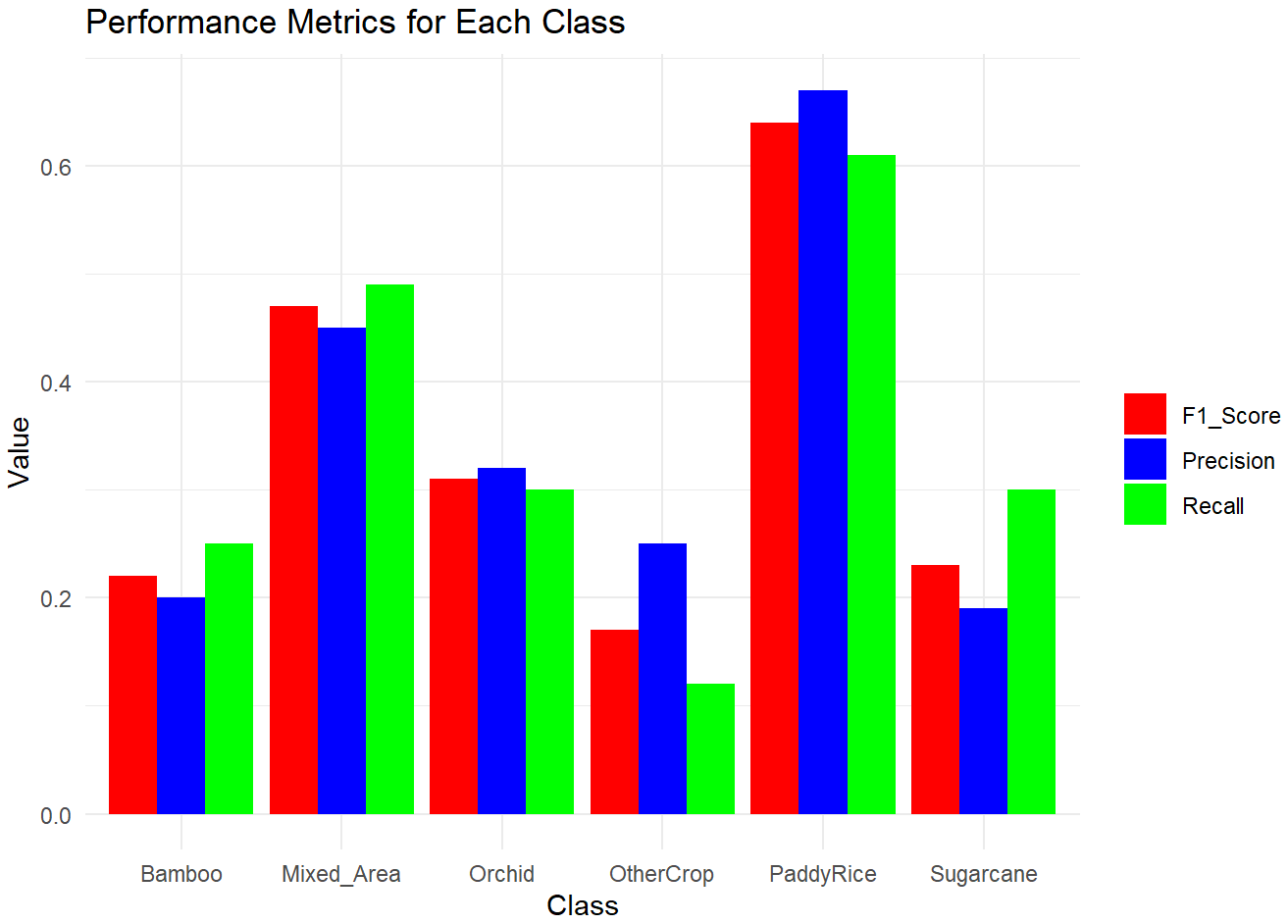
Precision, Recall, and F1-Score for Each Class

Class	Precision	Recall	F1_Score
Bamboo	0.20	0.25	0.22
Orchid	0.32	0.30	0.31
PaddyRice	0.67	0.61	0.64
Sugarcane	0.19	0.30	0.23
OtherCrop	0.25	0.12	0.17
Mixed_Area	0.45	0.49	0.47

```
# Transform the data for a grouped bar chart
results_long <- results_df %>%
  pivot_longer(cols = c(Precision, Recall, F1_Score), names_to = "Metric", values_to = "Value")

# Create a grouped bar chart for Precision, Recall, and F1-Score
grouped_bar <- ggplot(results_long, aes(Class, Value, fill = Metric)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Performance Metrics for Each Class", x = "Class", y = "Value") +
  scale_fill_manual(values = c("Precision" = "blue", "Recall" = "green", "F1_Score" = "red")) +
  theme_minimal() +
  theme(legend.title = element_blank())

# Display the grouped bar chart
print(grouped_bar)
```



Calculate Producer Accuracy (PA) and User Accuracy (UA)

- Created empty vectors to store producer accuracy (PA) and user accuracy (UA) values for each class.
- Calculated Producer Accuracy and User Accuracy for each class by dividing the true positive count by the sum of the actual counts for that class.
- Created Producer and User Accuracy Table and used the kable function to format the results as a table.
- Displayed the Producer Accuracy and User Accuracy value in form of Bar Diagram

```
# Initialize vectors to store producer accuracy (PA) and user accuracy (UA)
PA_values <- numeric(length(class_names))
UA_values <- numeric(length(class_names))


for (i in 1:length(class_names)) {
  class_name <- class_names[i]

  # Calculate Producer Accuracy (PA)
  PA <- confusion_matrix[class_name, class_name] / sum(confusion_matrix[class_name, ])

  # Calculate User Accuracy (UA)
  UA <- confusion_matrix[class_name, class_name] / sum(confusion_matrix[, class_name])



  # Store values in respective vectors
  PA_values[i] <- PA
  UA_values[i] <- UA
}

# Create a data frame for  Producer Accuracy and User Accuracy results
PA_UA_df <- data.frame(
  Class = class_names,
  Producer_Accuracy = round(PA_values, 2),
  User_Accuracy = round(UA_values, 2)
)

# Use kable to format the results as a table
kable(PA_UA_df, caption = "Producer and User Accuracy for Each Class")
```

Producer and User Accuracy for Each Class

Class	Producer_Accuracy	User_Accuracy
Bamboo	0.25	0.20
Orchid	0.30	0.32
PaddyRice	0.61	0.67
Sugarcane	0.30	0.19
OtherCrop	0.12	0.25
Mixed_Area	0.49	0.45

```
# Transform the data for a grouped bar chart for PA and UA
PA_UA_long <- PA_UA_df %>%
  pivot_longer(cols = c(Producer_Accuracy, User_Accuracy), names_to = "Metric", values_to = "Val
ue")

# Create a grouped bar chart for PA and UA
grouped_bar_PA_UA <- ggplot(PA_UA_long, aes(Class, Value, fill = Metric)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Producer and User Accuracy for Each Class", x = "Class", y = "Value") +
  scale_fill_manual(values = c("Producer_Accuracy" = "orange", "User_Accuracy" = "maroon")) +
  theme_minimal() +
  theme(legend.title = element_blank())

# Display the grouped bar chart for PA and UA
print(grouped_bar_PA_UA)
```