Project Title: GeoAI Challenge for Cropland Mapping

---

1. Executive Summary

This project aims to develop a cost-effective and accurate method for cropland mapping in arid regions using machine learning and remote sensing data. Using time-series satellite imagery (Sentinel-1 and Sentinel-2), the project leverages geospatial features and a Random Forest model to classify cropland vs non-cropland areas in **Fergana (Uzbekistan)** and **Orenburg (Russia)**. The final model achieves high accuracy while being computationally efficient and reproducible using open-source tools.

---

2. Problem Statement

Mapping cropland accurately in arid and semi-arid regions is challenging due to **spectral similarities with pastures and steppe land**. This project addresses the challenge using **open-source tools** and **public satellite imagery** to distinguish between cropland and other land types for sustainable land use monitoring.

---

3. Data Description

The dataset includes **Sentinel-1** and **Sentinel-2** time-series satellite imagery.

- Each entry includes a spatial location and corresponding multi-band reflectance values.
- The **target variable** is a binary label:
  - **1** = cropland
  - **0** = non-cropland

---

4. Methodology

- Data was preprocessed to remove irrelevant columns and merged across sensors.
- Time-series features such as **NDVI**, **SAVI**, and raw bands were used.
- Data imbalance was addressed using **SMOTE** oversampling.
- Features were scaled using **StandardScaler** or **RobustScaler**.

- A **Random Forest Classifier** was trained with hyperparameter tuning using **GridSearchCV**.
- Accuracy and F1 scores were calculated on validation data.

---

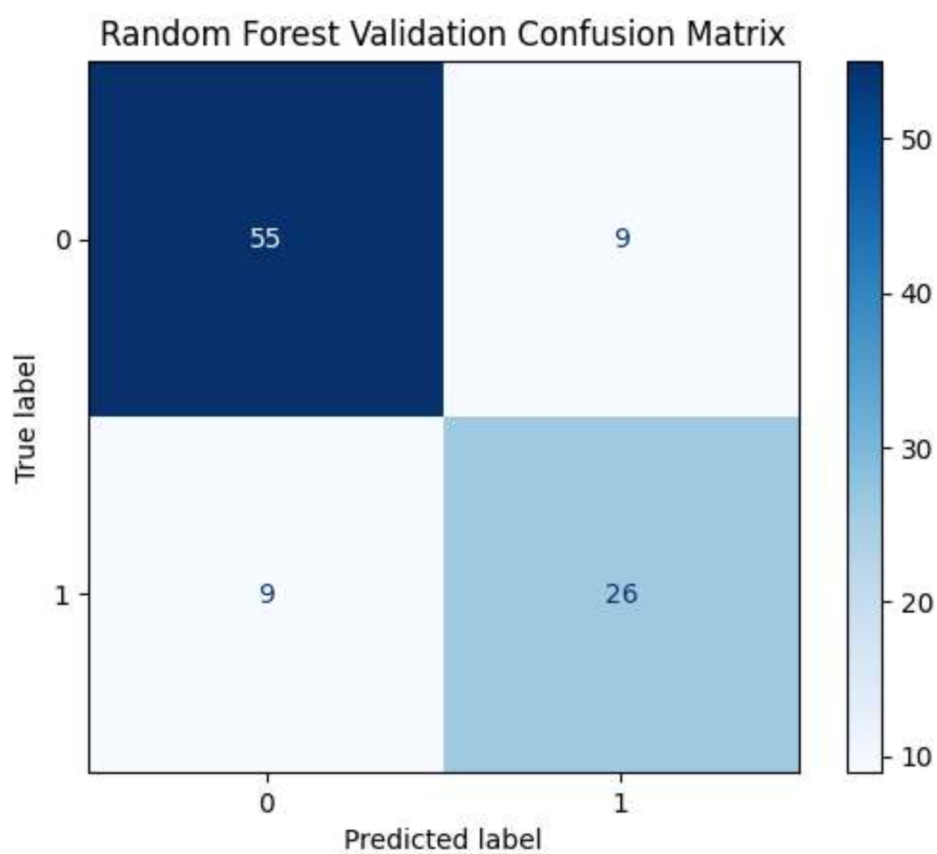
5. Results

The final Random Forest model achieved:

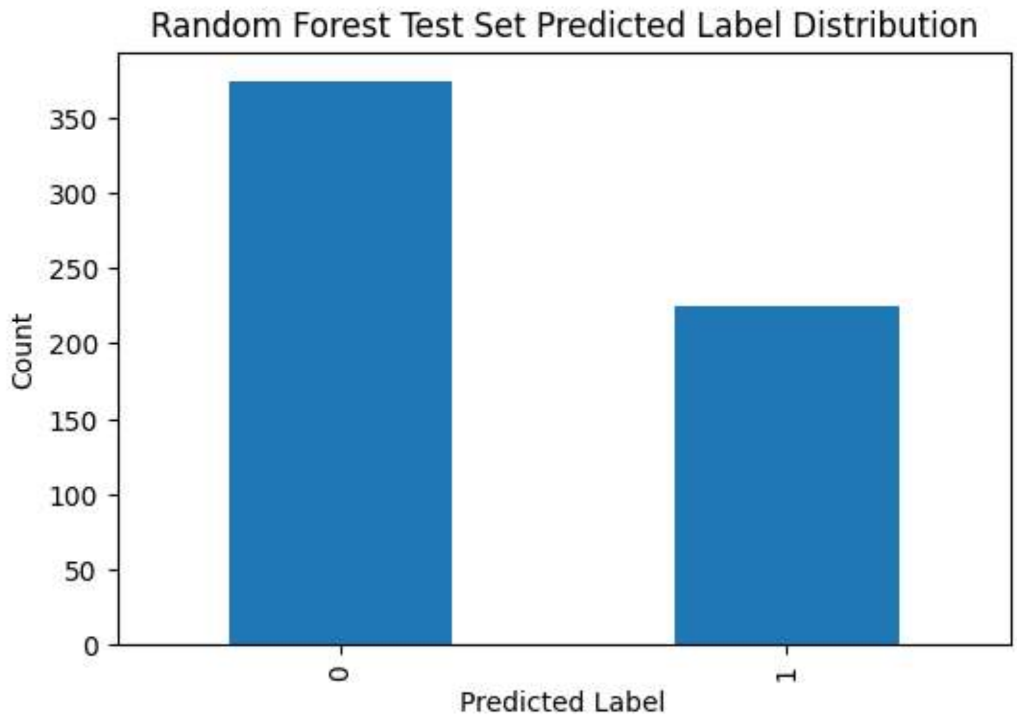
- **High classification accuracy**
- **Balanced F1 Score**

```
Random Forest Validation Accuracy: 0.8181818181818182
Random Forest F1 Score: 0.8181818181818182
```

- A well-distributed **confusion matrix** with minimal false positives and false negatives.



These results confirm the effectiveness of using ensemble models with engineered geospatial and temporal features for cropland mapping in dry regions.



6. Recommendations and Future Work

- Explore additional vegetation indices such as **EVI**, **NBR**, and **NDWI** for better representation.
- Include **weather**, **soil**, and **elevation data** to improve the temporal and spatial context.
- Experiment with **deep learning models** (e.g., LSTM, 1D CNN) to learn from longer time series.
- Improve spatial generalization by training on **diverse agro-ecological zones**.

---

7. Appendix

- **Libraries used:** scikit-learn, imbalanced-learn, geopandas, matplotlib, seaborn, pyproj
- **Environment:** Python 3.10+, runs on Google Colab or local machines (8GB RAM minimum)
- **Expected Runtime:** 5–15 minutes per model training iteration

- **Submission Format:**
  A CSV file with two columns:

ID, Target

ID_C7AV4GEJP9, 1

ID_AFVZYGLXXY, 0

8. Application Workflow, Challenges, Cost & Farmer Outreach

8.1. How the App Works

The cropland mapping solution can be converted into a lightweight web or Android app with the following workflow:

1. **User Input:**
   - User selects or uploads satellite data (e.g., via coordinates or location name).
   - Optionally, users can upload their own satellite band CSV data.
2. **Backend Processing:**
   - The app runs the trained Random Forest model.
   - It processes the data through feature engineering pipelines.
   - It predicts whether each point is **cropland** or **non-cropland**.
3. **Results Visualization:**
   - Predicted cropland areas are shown on an interactive map.
   - Users can download the results in CSV or shapefile format.
4. **Additional Features (Optional):**
   - Crop rotation alerts (based on historical imagery)
   - Rainfall forecasts (via weather API)
   - Fertilizer or soil recommendations

Tech Stack for App Deployment:

- **Frontend:** React Native or Flutter

- **Backend:** Python + FastAPI or Flask
- **Hosting:** Firebase or AWS
- **Model Serving:** Streamlit, Gradio, or TensorFlow Lite

8.2. Development Challenges

Some of the challenges encountered include:

- **Spectral confusion** between cropland and natural vegetation in dry seasons.
- **Sparse labeled data** in arid zones, which limits training accuracy.
- **Cloud cover** in optical imagery required fallback to radar data (Sentinel-1).
- **Internet dependency** in remote regions for app usage.
- Ensuring the **model generalizes** well across countries and soil types.

8.3. Cost of Publishing the App on Google Play Store

Item	Estimated Cost (USD)
Google Play Developer Account (One-time fee)	$25
Backend Hosting (Firebase, 3 months free, then...)	$5–10/month
Domain Name & SSL (optional)	$10–15/year
UI/UX Design (basic prototype)	$50–100 (freelancer)
Total Initial Budget (MVP)	**$90–150**

8.4. Marketing Strategy to Reach Farmers in Arid/Semi-Arid Regions

To make real impact, the app must be positioned where it's needed most — remote areas with agricultural activity and data gaps.

Channels:

- 📱 **WhatsApp & SMS Alerts:** Leverage existing farmer groups to share download links and explain benefits in local languages.

- 👨🏾‍🌾 Farmers' Cooperatives & Extension Officers: Partner with rural cooperatives and government extension workers to promote the tool.
- 📻 **Local Radio Campaigns:** Short educational jingles on how the app helps farmers make data-driven decisions.
- 🛰️ **NGO/Government Partnerships:** Collaborate with FAO, WFP, or national agriculture bodies to scale reach.

Messaging:

- "📍 Map Your Land. Know Your Crop."
- "🌾 Get Instant Cropland Analysis from Satellite — Free for Farmers!"
- "🚜 No Internet? Use Offline Mode and Still Get Results."