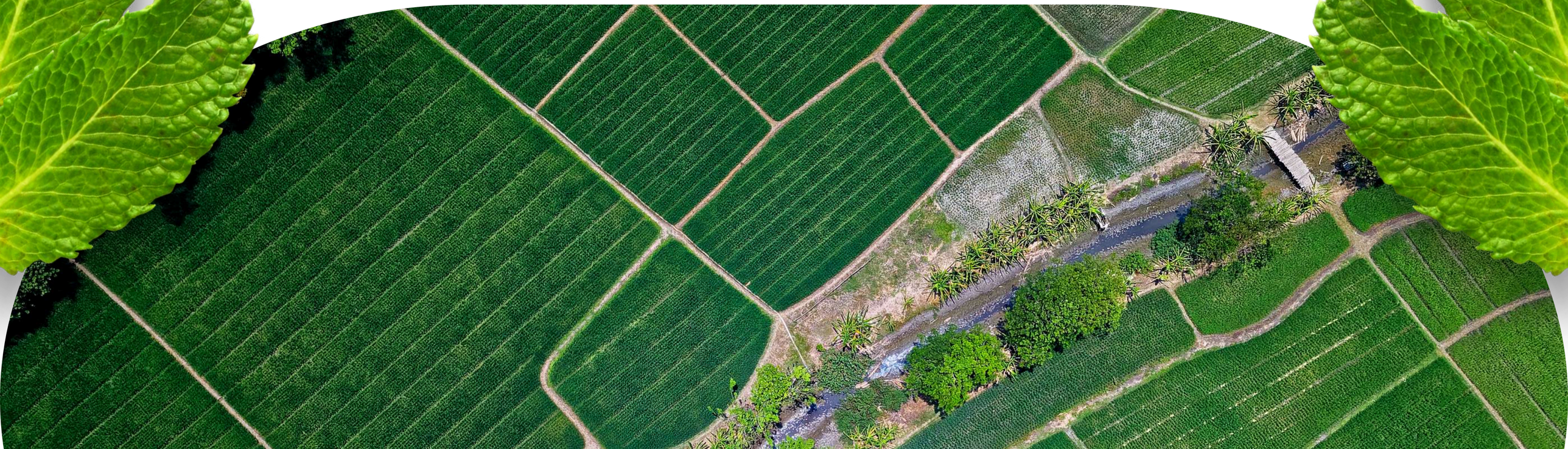
CROP HEALTH ANALYSIS

Team 18: Anant Jain, Bhavya Jain 1, Sahil Dhiman, Sohan Panda



Background

- By 2050, it is predicted that there will be between 8.0 and 10.4 billion people on earth, with a median value of 9.1 billion. If all of these people are to be fed sufficiently, total food consumption will have to increase by 50-70% (Ober,2010).
- Developing food security and declining hunger by 2050 are beneficial critical objectives for the United Nations. Hence crop protection and land assessment are of more considerable significance to global food production.
- A staggering 33% of crop yield loss in India is caused by **biotic stress**, which is a major constraint in crop production. Among the major **pests**, weeds cause 12.5% loss, whereas insects in the field inflict 9.5% loss, **diseases** 6.5% and other pests 4.5% loss (DWR 2015).

Indian Council of Agricultural Research
https://www.jstor.org/stable/20752982?seq=1

Problem Statement

Crop health monitoring is crucial for early detection of diseases, pest infestations, and environmental stress. However, existing methods are limited by scalability, high costs, real-time adaptability, and accessibility for small-scale farmers. There is a need for an AI-driven solution that provides real-time crop health analysis, identifying areas affected by stress, pests, and diseases, enabling farmers to take timely action for improved productivity and sustainability.

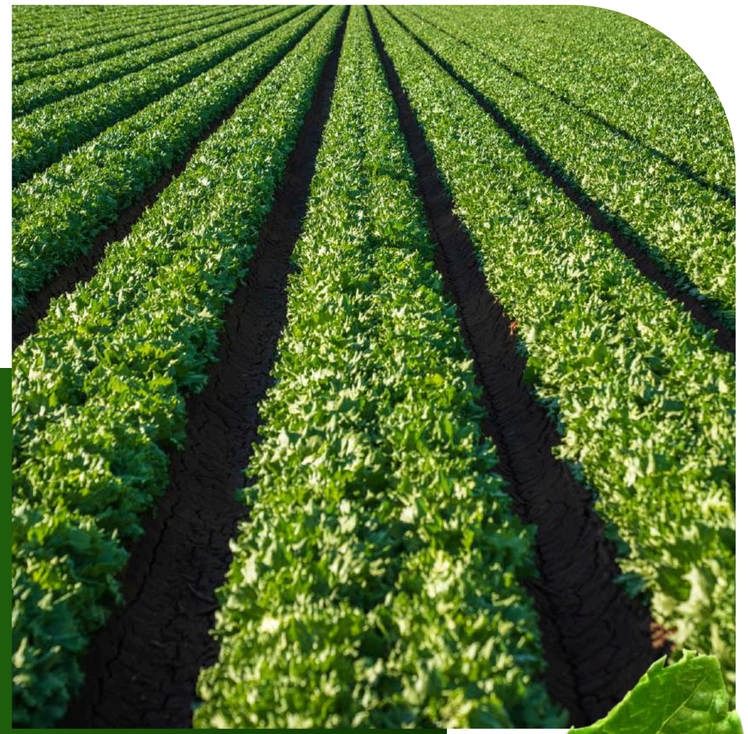
Research Questions

- How can satellite imagery and temporally sensitive weather patterns be used to build a scalable system for detailed crop health monitoring across diverse crop types?

- Can a four-class classification framework—categorising crops as healthy, stressed, diseased, or pest-infested—provide more actionable insights for precision agriculture than traditional binary classification models?

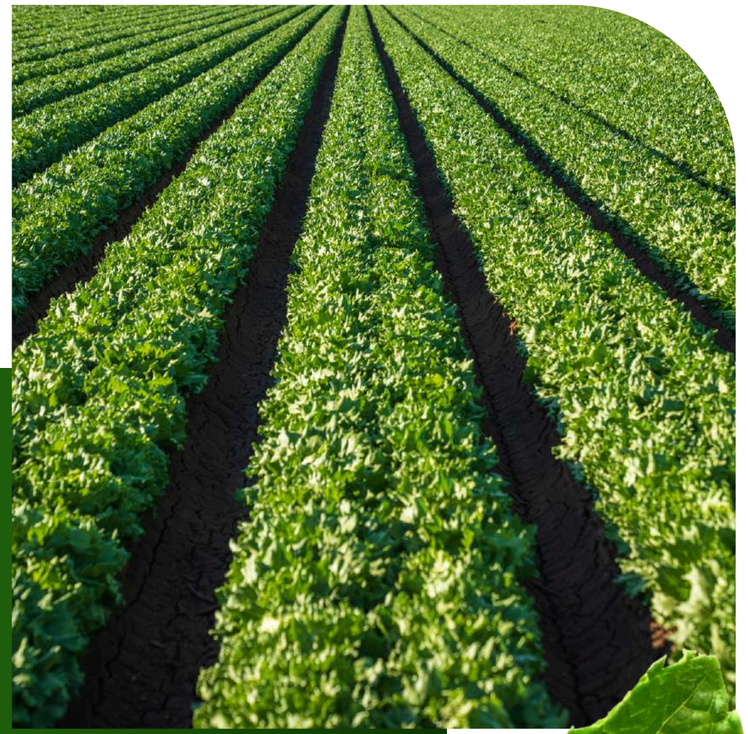
- How do machine learning and deep learning models compare in performance when trained on integrated satellite and meteorological data for multi-class crop health classification?

Potential Applications



- **Early Detection** – Identifies crop diseases, pests, and stress before significant damage occurs.
- **Precision Farming** – Enables targeted treatment, reducing input costs and improving efficiency.
- **Remote Monitoring** – Uses ML with satellite or drone imagery for large-scale crop health assessment.
- **Optimized Resource Use** – Helps manage irrigation, fertilizers, and pesticides effectively.

Impact



In India's economy, for the people who are living in rural areas, agriculture is the primary occupation of more than half of the population, but it only accounts for 17% of the country's GDP, according to 2018 statistics.

By improving crop health and reducing losses through AI-driven solutions, we aim to enhance agricultural productivity, ultimately increasing its share in the economy.

S. Iniyan, V. Akhil Varma, and C. Teja Naidu, “Crop yield prediction using machine learning techniques,” Advances in Engineering Software, vol. 175, p. 103326, Jan. 2023, doi: 10.1016/j.advengsoft.2022.103326.

Literature Review 1

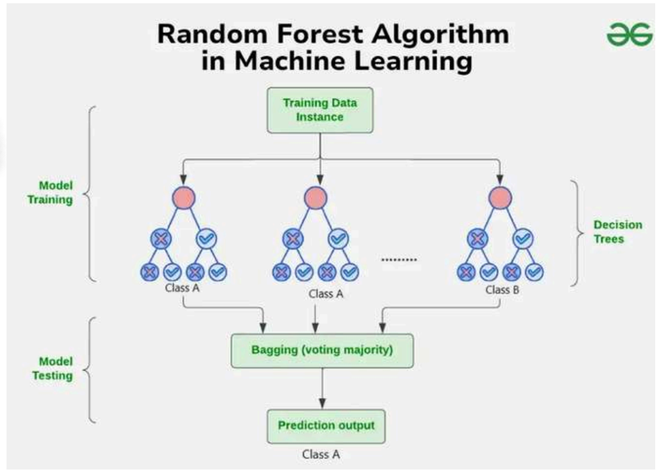
2.3.2. Random Forest

Random Forest (RF) is a widely known ensemble built from decision trees trained on different subsets of the training data. Also, when deciding which variable to split on a node, RF considers a random set of variables and not the whole set of features. During classification, each tree votes and the class most agreed upon is returned. As each tree is trained on a subset of data and of features, the computation is fast. A high number of trees and the diversity of each of them makes them robust to noise and outliers. Some studies that have employed Random Forest (RF) are shown in Table 4.

Table 4. Performance of Random Forests.

Classification/Regression	Number of Trees	Performance
Regression	100	$r^2 = 0.75$
Regression	200	$r^2 = 0.75$
Classification	-	70.0% acc.
Classification	-	95.5% acc., 94.2% f1

RFs can achieve greater accuracy with less number of samples when compared to other ML techniques [77].



Extreme Gradient Boosting (XGBoost) is a more advanced ensemble method than Random Forest known for its superior predictive performance and computational efficiency (Fatima et al., 2023). **Unlike RF, XGBoost builds trees sequentially, with each new tree learning to correct errors of the previous ones**. This gradient boosting approach enables XGBoost to capture subtle patterns and minimise bias more effectively, especially in datasets with complex interactions and class imbalance.

Fatima, S., Hussain, A., Amir, S. B., Ahmed, S. H., & Aslam, S. M. H. (2023). XGBoost and random forest algorithms: an in depth analysis. Pakistan Journal of Scientific Research, 3(1), 26-31.

Literature Review 1 Interpretation

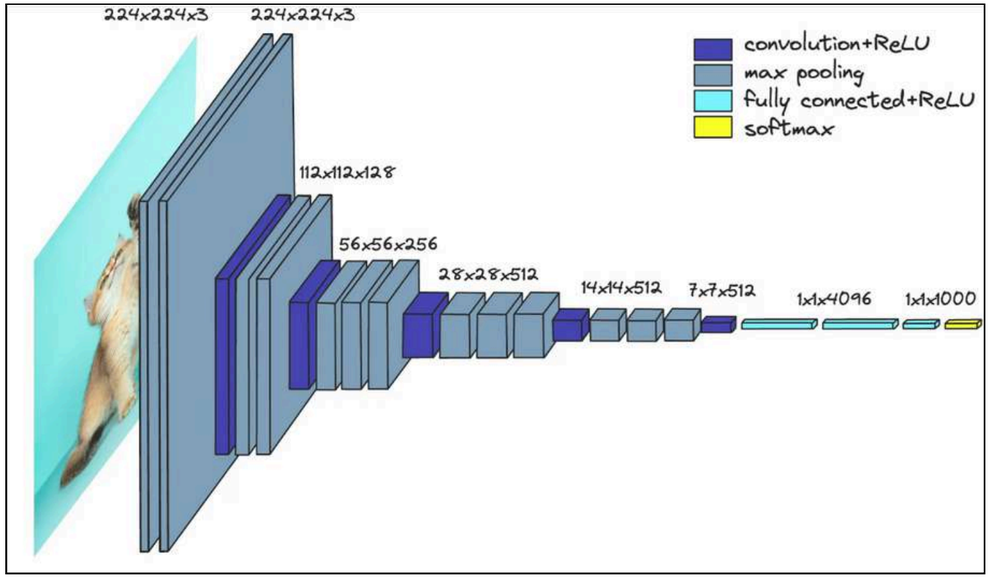
Objective: The paper is trying to predict crop diseases and pest, for tomato plant.
Dataset: There dataset has been captures both manually in greenhouses and taken from satellites.

Limitations: XGBoost and Random Forest, while effective for structured tabular data, are not ideal for high-dimensional image inputs. They lack the ability to capture spatial relationships and temporal patterns inherent in image sequences. As a result, applying these models to crop health classification requires manual extraction of features like vegetation indices, which can miss crucial contextual information.

In our case, due to the significant class imbalance in our datase, primarily healthy crop samples, and the nature of the features provided by Zindi, Random Forest and XGBoost performed reasonably well in identifying only Healthy fields, even when we integrated vegetation indices. However, both models struggled to accurately classify other crop health categories such as diseased or pest-infested crops.

Literature Review 2

In this study, a Convolutional Neural Network (CNN) model with custom dense layers was developed for accurate and efficient crop disease detection. The model made use of three primary convolutional layers and two dense layers to increase the efficiency of the model. This is different when compared to the earlier studies as instead of using a large number of convolutional layers, this model prioritizes speed so that farmers can be notified about their diseases as soon as possible. The model's efficacy was evaluated using three diverse



Rajvanshi, A. (2024, February 23). Early detection of crop diseases using CNN Classification - NHSJS. NHSJS. https://nhsjs.com/2024/early-detection-of-crop-diseases-using-cnn-classification/

Literature Review 2 Interpretation

Dataset: The study used image datasets for **apple, corn, and tomato crops**, combining:
PlantVillage dataset (Kaggle): High-quality, labeled images of healthy and diseased leaves.
Field images (Gujarat farms): Real-world photos captured under varied conditions and verified by an agricultural expert.

Classes per crop:
Apple: Healthy, Apple Scab, Apple Cedar Rust
Corn: Healthy, Northern Blight, Common Rust
Tomato: Healthy, Early Blight, Late Blight

Limitations:
Limited Generalizability: Trained on specific locations; may not perform well in new environments.
Low Class Diversity: Only detects a few diseases; no pest or stress detection.
Labeling Errors: Potential mislabeling, especially for similar-looking diseases.
Image Variability: Sensitive to lighting, background, and angles.
Overfitting Risk: Insufficient data for rare classes.
No Localization: Cannot detect disease spots or handle multiple leaves/diseases.

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. Computers and Electronics in Agriculture, 147, 70–90. https://doi.org/10.1016/j.compag.2018.02.016

Literature Review 3

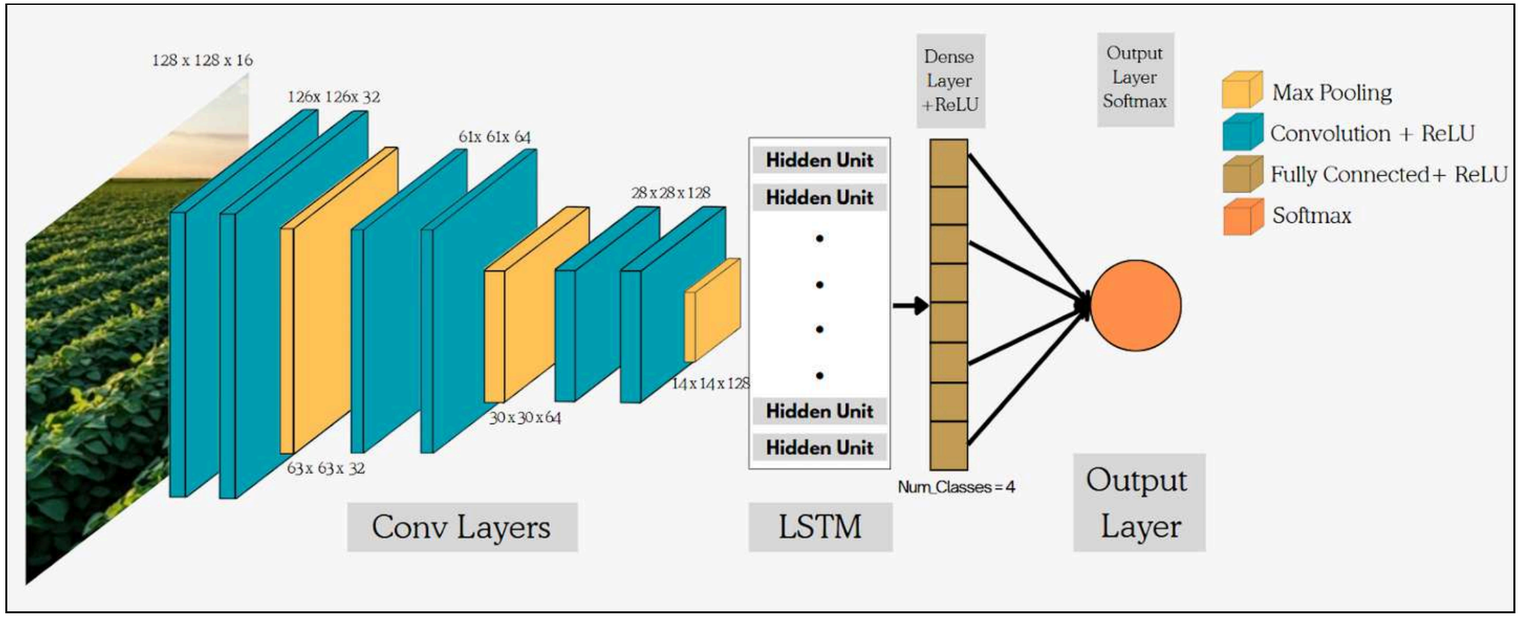
Amado emphasizes that **LSTM neural networks** are prepared for receiving sequential data as input and are able to extract important aspects related to the time series since it maintains a chain structure with time steps, **similar to the way that crop growth modeling works**. Each step takes information from previous steps and outside input (from feature space – **new NDVI, EVI, LST and precipitation values**), and provides output for the next step. Furthermore, during the training process this algorithm is capable of retaining key information of input signals, ignoring less important parts. These models can process sequential data—like **canopy change over time**—and recognise latent interactions that impact crop development.

Convolutional Neural Networks (CNNs) and **Long Short-Term Memory (LSTM)** networks have become especially prevalent in **agricultural image analysis and time-series forecasting**. Prenafeta-Boldú (2018) highlights that **DL-based models** can efficiently deal with **raster-based data (e.g. video, images)**, and thus can be used to analyse pictures of the crop field for classification. It can also be applied to any form of data, such as audio, speech, and natural language, or more generally to continuous or single point data such as **weather data** (Sehgal, et al., 2017), soil chemistry (Song, et al., 2016) and much more.

Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. Computers and Electronics in Agriculture, 147, 70–90. https://doi.org/10.1016/j.compag.2018.02.016

Literature Review 3 Interpretation

- CNNs effectively extract spatial features from multispectral and hyperspectral images, enabling detection of weeds, stress, diseases, and pests.
- LSTMs capture temporal dependencies from sequential data such as vegetation indices and weather variables, identifying deviations from normal crop growth patterns.
- This integration allows for the detection of early anomalies in crop health, enabling dynamic, data-driven decision-making.

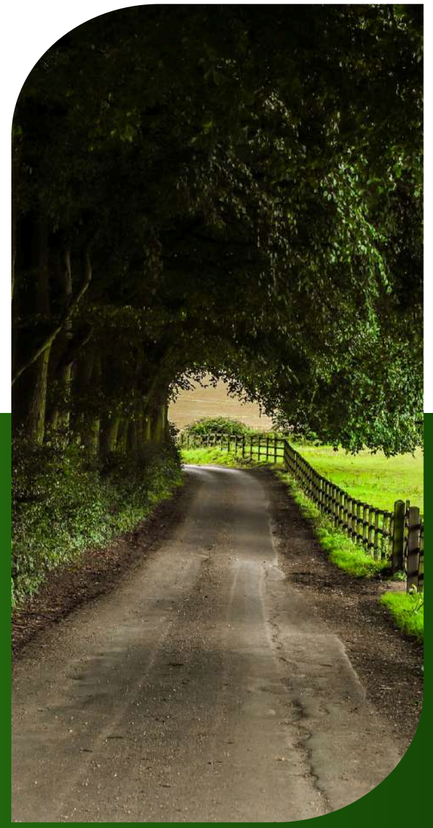


Literature Review 4

Thus, considering the importance of soybean in Brazil and its impact on the global economy, and the evident lack of reliable yield information in near real-time basis, the implementation of a near-real time yield forecast will provide a useful layer for agricultural purposes and policy applications. Therefore, the objectives of this research were to: i) compare the performance of three different algorithms (multivariate ordinary least square – OLS - linear regression, random forest and LSTM neural network) for forecasting soybean yield using vegetation indices such as NDVI, EVI, and weather data such as land surface temperature and precipitation as independent variables, and ii) evaluate how early (during the soybean growing season) this method is able to forecast yield with reasonable accuracy.

R. A. Schwalbert, T. Amado, G. Corassa, L. P. Pott, P. V. V. Prasad, and I. A. Ciampitti, "Satellite-based soybean yield forecast: Integrating machine learning and weather data for improving crop yield prediction in southern Brazil," Agricultural and Forest Meteorology, vol. 284, p. 107886, Apr. 2020, doi: 10.1016/j.agrformet.2019.107886.

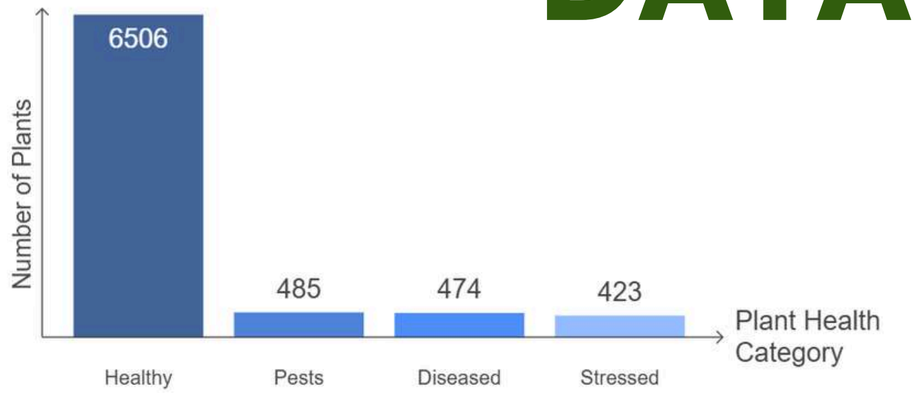
GAP



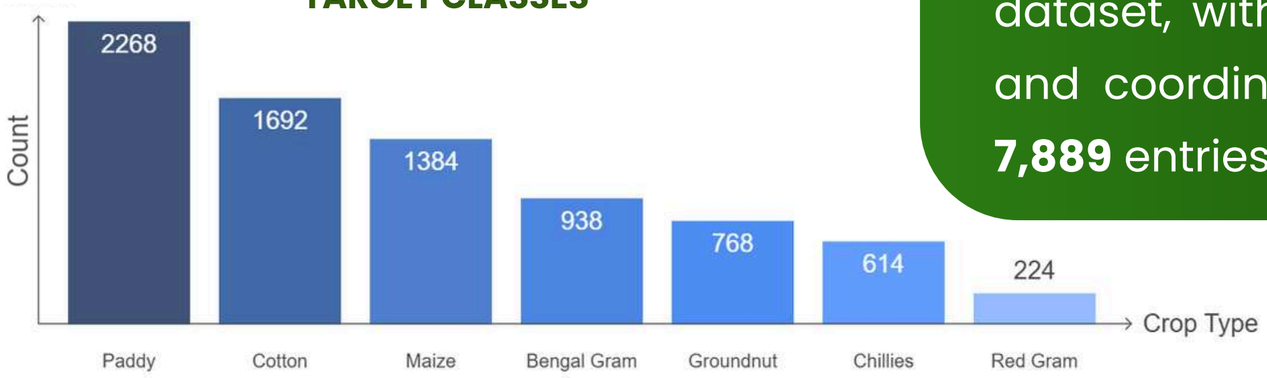
- There is a need for a scalable system that integrates satellite imagery with temporally sensitive weather data for detailed crop health monitoring across diverse crop types.
- The effectiveness of a four-class classification framework (healthy, stressed, diseased, pest-infested) compared to traditional binary models remains underexplored.
- Limited comparative analysis exists on the performance of machine learning versus deep learning models when trained on integrated satellite and meteorological data for multi-class crop health classification.
- A recent study by Javadinejad et al. identified a correlation between reduced crop yields and two environmental factors: elevated temperatures and increased precipitation. However, the integration of temperature and rainfall data in crop health monitoring models is often limited or missing.

https://ieeexplore.ieee.org/abstract/document/10485420

DATASET SOURCE



TARGET CLASSES



Crop Count Distribution

- Source: Dataset obtained from an **African ML competition** hosted by **Zindi**, a platform for data science challenges.
- Relevance: The dataset includes real-world crop health labeled dataset, with 4 categories- Stressed, Pests, Healthy, Diseased; and coordinates of the crop fields in Telangana. (**Train Set- 7,889** entries and **Test Set- 2719** entries)

	State	District	Sub-District	SDate	HDate	CropCoveredArea	CHeight	CNext	CLast	CTransp	IrriType	IrriSource	IrriCount	WaterCov	ExpYield	Season	geometry
2	Telangana	Medak	Kulcharam	25-11-2023 00:00	14-04-2024 00:00	97	54	Pea	Lentil	Transplan	Flood	Groundwat	4	87	17	Rabi	POLYGON ((78.181432460760
3	Telangana	Medak	Kulcharam	13-11-2023 00:00	26-04-2024 00:00	82	58	Pea	Lentil	Transplan	Flood	Canal	5	94	15	Rabi	POLYGON ((78.175451775474
4	Telangana	Medak	Kulcharam	19-12-2023 00:00	28-04-2024 00:00	92	91	Pea	Lentil	Transplan	Flood	Canal	3	99	20	Rabi	POLYGON ((78.169142247707
5	Telangana	Medak	Kulcharam	11-02-2023 00:00	04-11-2024 00:00	91	52	Pea	Lentil	Transplan	Flood	Canal	5	92	16	Rabi	POLYGON ((78.168891038419
6	Telangana	Medak	Kulcharam	12-12-2023 00:00	19-05-2024 00:00	94	55	Pea	Lentil	Transplan	Flood	Canal	5	97	20	Rabi	POLYGON ((78.172644529980
7	Telangana	Medak	Kulcharam	13-12-2023 00:00	18-05-2024 00:00	97	51	Pea	Lentil	Transplan	Flood	Groundwat	5	85	15	Rabi	POLYGON ((78.172990949787
8	Telangana	Medak	Kulcharam	20-11-2023 00:00	30-05-2024 00:00	84	68	Lentil	Pea	Transplan	Flood	Canal	3	90	15	Rabi	POLYGON ((78.176543766873
9	Telangana	Medak	Kulcharam	14-12-2023 00:00	04-04-2024 00:00	86	72	Pea	Lentil	Transplan	Flood	Canal	3	91	15	Rabi	POLYGON ((78.178821495411
10	Telangana	Medak	Kulcharam	12-10-2023 00:00	05-10-2024 00:00	90	78	Pea	Lentil	Transplan	Flood	Canal	3	86	18	Rabi	POLYGON ((78.181035872335
11	Telangana	Medak	Kulcharam	12-10-2023 00:00	22-04-2024 00:00	90	53	Lentil	Pea	Transplan	Flood	Groundwat	5	90	19	Rabi	POLYGON ((78.180790888349
12	Telangana	Medak	Kulcharam	12-02-2023 00:00	29-05-2024 00:00	94	56	Lentil	Pea	Transplan	Flood	Canal	4	91	17	Rabi	POLYGON ((78.178809777193

https://zindi.africa/competitions/telangana-crop-health-challenge?ref=mlcontests

FEATURES PREPROCESSING

1. Extracted satellite images using Google Earth Engine
2. Added vegetation indices
3. Removed null values
4. Used label encoding
5. Generated correlation map
6. Reformatted meteorological data
7. Standarization
8. Average Temp, Humidity and Rainfall during the growing phase
9. Time series satellite images

https://data.telangana.gov.in/dataset/telangana-weather-data-2023-2024

https://zindi.africa/competitions/telangana-crop-health-challenge?ref=mlcontests

Features Preprocessing

Extracted Satellite Image Using Google Earth Engine

We extracted the most recent image available within the specified time frame, geometry and cloud cover constraints from Google Earth Engine.

```python
def download_rgb_image(collection_name, bands, start_date, end_date, region, output_folder='downloads'):
    """Download RGB bands from a GEE collection filtered by date and region."""
    # Load the image collection, filter by date, and clip to region
    collection = ee.ImageCollection(collection_name).filterDate(start_date, end_date).filterBounds(region)
    image = collection.sort('system:time_start', False).first().select(bands).clip(region)  # Most recent image

    # Define unique filename based on image dates
    image_id = image.id().getInfo() or f'image_{start_date}_{end_date}'
    image_name = f'{output_folder}/{image_id}_RGB_{start_date}_{end_date}.tif'

    # Export the image to a GeoTIFF file
    geemap.ee_export_image(
        image,
        filename=image_name,
        scale=10,  # Sentinel-2 resolution in meters
        region=region,
        file_per_band=False,  # Save as a multi-band TIFF
        crs='EPSG:4326'
    )
    print(f"Downloaded: {image_name}")
    return image_name
```

Additionally, we retrieved a time series of images by selecting 5 images of each crop sample, equally spaced over their growth periods , which allowed us to monitor temporal changes in vegetation and crop health throughout the growing season.

FEATURES PREPROCESSING

We have enhanced our dataset by integrating meteorological data with existing crop data, generating new features such as **average temperature** and **rainfall** to improve crop health analysis. (Telangana Govt.)

We have further enriched our dataset by incorporating agricultural indices such as **NDVI, EVI, MSAVI, and GNDVI,** calculated using satellite imagery to derive valuable insights into crop health.

https://data.telangana.gov.in/dataset/telangana-weather-data-2023-2024

https://zindi.africa/competitions/telangana-crop-health-challenge?ref=mlcontests

Features Preprocessing

Average Temp and Rainfall

Collected daily temperature, rainfall, and humidity data from the official Telangana government website, organized at the district and sub-district levels.

We computed the average rainfall over the crop's lifecycle, from sowing to harvesting.
Moreover, we also calculated the minimum and maximum average temperatures and humidity during the same period.

District	Mandal	Date	Rain (mm)	Min Temp (	Max Temp	Min Humid	Max Humic	Min Wind S	Max Wind S
Adilabad	Adilabad R	01-May-24	0	25.3	43.5	31	59.2	0.3	9
Adilabad	Adilabad R	02-May-24	0	23.8	42.8	22	47.9	0	8.8
Adilabad	Adilabad R	03-May-24	0	21.3	42.8	15.7	44.9	0	11.2
Adilabad	Adilabad R	04-May-24	0	23.6	43.9	15.1	40	0	9.7
Adilabad	Adilabad R	05-May-24	0	24.4	44.1	17.5	44.4	1.4	14.2
Adilabad	Adilabad R	06-May-24	0	26.8	45.8	20	55.2	0	10.5
Adilabad	Adilabad R	07-May-24	0	30.1	44.3	24.8	68.2	0	10.7
Adilabad	Adilabad R	08-May-24	0.9	24.1	40.7	40.3	77.3	4.8	21.3
Adilabad	Adilabad R	09-May-24	0	29.4	41.2	33.6	66.2	0.3	17.8
Adilabad	Adilabad R	10-May-24	0.2	26	40.9	34.3	72.8	0	21.4
Adilabad	Adilabad R	11-May-24	0	28.8	41.8	34.4	63.7	2.8	14.7
Adilabad	Adilabad R	12-May-24	0	29.4	40.9	34.3	65.7	0	14.8
Adilabad	Adilabad R	13-May-24	17	22.9	41.3	35.3	91.2	0	18.8
Adilabad	Adilabad R	14-May-24	0	25.2	35.1	46	75.9	0	9.7
Adilabad	Adilabad R	15-May-24	2.5	23	39.8	37.6	95.1	0	6.9
Adilabad	Adilabad R	16-May-24	0	25.4	35	50.7	82.1	0	9.9
Adilabad	Adilabad R	17-May-24	0	24.7	38.3	47	83	1.3	15
Adilabad	Adilabad R	18-May-24	0	25.3	35.9	46.6	83.5	0	10.3
Adilabad	Adilabad R	19-May-24	0	27.2	39.2	43.9	81.3	1.6	15.9
Adilabad	Adilabad R	20-May-24	0	28.7	41.1	39.4	62.7	1.4	12.1
Adilabad	Adilabad R	21-May-24	0	30.5	40.8	34.1	67.4	1.1	9.8
Adilabad	Adilabad R	22-May-24	0	30.4	41.3	32.6	65.1	1.5	12.8
Adilabad	Adilabad R	23-May-24	5.2	26	41.7	37.9	82.8	1	8.9

Features Preprocessing

Vegetation Indices

AE	AF	AG	AH	AI	AJ
NDVI	GNDVI	CIRE	NDRE	PRI	MSAVI
0.097891	0.060872	0.149639	0.064787	-0.00401	0.164567
0.153496	0.225163	0.202944	0.088903	0.140486	0.254773
0.186761	0.237447	0.251967	0.107416	0.134588	0.303626
0.182659	0.203138	0.280396	0.110882	0.098446	0.275923
0.085718	0.149046	0.10458	0.048818	0.101394	0.154427
0.109498	0.15954	0.132785	0.061048	0.099827	0.192727
0.05706	0.078259	0.010606	0.005243	0.073057	0.107059
0.327901	0.303047	0.727977	0.239256	0.073175	0.451036
0.061671	0.045578	0.063419	0.030175	0.015448	0.112835
-0.0854	-0.18533	-0.1167	-0.06293	-0.1245	-0.19086
0.055444	0.057788	0.054064	0.02582	0.032107	0.101249
0.237447	0.214527	0.41717	0.15327	0.067853	0.33695
0.062235	0.090784	0.071759	0.033486	0.057763	0.112566
0.16937	0.236645	0.235962	0.102907	0.137708	0.282994
0.125228	0.170417	0.17105	0.075362	0.09801	0.209662
0.105696	0.174685	0.132497	0.059557	0.117957	0.180877
0.183583	0.232086	0.252021	0.107248	0.130107	0.295229
0.255051	0.288073	0.345641	0.143929	0.151949	0.395778
0.018759	-0.00699	0.008479	0.003698	-0.01069	0.035699
0.121901	0.193252	0.151777	0.069482	0.125988	0.213045
0.068539	0.074103	0.12646	0.058979	0.01521	0.127136
0.10632	0.099385	0.157789	0.071321	0.028422	0.186139
0.033411	0.031057	0.020375	0.009647	0.021417	0.062996

1. **NDVI (Normalized Difference Vegetation Index)**

$$\text{NDVI} = \frac{NIR - RED}{NIR + RED}$$

2. **GNDVI (Green Normalized Difference Vegetation Index)**

$$\text{GNDVI} = \frac{NIR - GREEN}{NIR + GREEN}$$

3. **CIRE (Chlorophyll Index Red Edge)**

$$\text{CIRE} = \frac{NIR}{REDEDGE} - 1$$

4. **NDRE (Normalized Difference Red Edge Index)**

$$\text{NDRE} = \frac{NIR - REDEDGE}{NIR + REDEDGE}$$

5. **PRI (Photochemical Reflectance Index)**

$$\text{PRI} = \frac{R_{531} - R_{570}}{R_{531} + R_{570}}$$

6. **MSAVI (Modified Soil Adjusted Vegetation Index)**

$$\text{MSAVI} = \frac{2 \cdot NIR + 1 - \sqrt{(2 \cdot NIR + 1)^2 - 8 \cdot (NIR - RED)}}{2}$$

Features Preprocessing

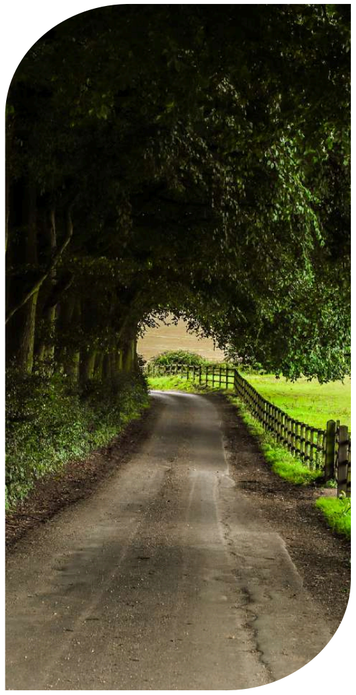
Standardisation

```
numeric_cols = ["WaterCov",'ExpYield', 'IrriCount','CropCoveredArea', 'CHeight','ndvi','evi','ndwi','gndvi','savi','msavi','Avg_Temperature','rainfall']
scaler = StandardScaler()
scaled_data = scaler.fit_transform(data[numeric_cols])
scaled_data
```

```
array([[ 1.35845477,  0.26405193,  0.32806677, ...,  0.26924472,
        -0.93582983, -0.80248173],
       [ 1.66821601, -0.09739723,  1.02806338, ..., -0.38436185,
        -0.69337113, -0.80248173],
       [ 1.88947404,  0.80622567, -0.37192984, ..., -0.05691935,
        -1.4587208 , -0.80041972],
       ...,
       [ 0.07515822, -0.27812181, -0.37192984, ..., -0.27356865,
         0.79977061,  0.94863287],
       [-0.19035142,  0.80622567, -0.37192984, ...,  3.86359403,
         1.82633135,  1.9025059 ],
       [-0.45586105, -1.18174471, -1.07192645, ..., -0.15078358,
         0.69403504, -0.55178251]])
```

Features Preprocessing

Handling NAN Values



W	X	Y	Z
AvgMinHumidit	AvgMaxHumidity(%)	Min Temp Av	Max Temp Avg
36.62447552	89.46853147	18.52	33.44
36.42814371	89.11976048	18.84	33.83
33.13030303	86.61439394	19.43	34.76
47.11306376	90.30186625	21.29	33.82
32.746875	85.58375	19.87	35.33
32.6164557	85.46265823	19.88	35.35
35.09226804	86.36185567	20.04	35.08
34.0079646	89.66637168	18.2	33.64
47.23646409	90.7621547	21.02	33.69
37.78214286	90.73418367	18.62	33.52
40.94099379	88.45569358	20.7	34.44
47.90793651	91.78793651	20.18	32.57
38.35220884	92.7124498	19.9	33.7
35.85	100	14.3	30.95
32.07022901	90.13816794	19.84	34.49
32.668	91.2288	19.51	34.04
34.30916031	85.54274809	18.79	33.35
42.88752475	92.75564356	20.62	33.35
31.76493506	88.12597403	20.79	35.44
34.34759615	89.44326923	20.39	34.67
42.2804878	88.57272727	21.02	34.36

```python
# Basic feature setup
categorical_features = ['State', 'District', 'Crop', 'Season']
numeric_features = ['SowingMonth', 'HarvestMonth', 'CropDuration(Days)']

preprocessor = ColumnTransformer(transformers=[
    ('cat', Pipeline([
        ('imputer', SimpleImputer(strategy='most_frequent')),
        ('encoder', OneHotEncoder(handle_unknown='ignore'))
    ]), categorical_features),
    ('num', Pipeline([
        ('imputer', SimpleImputer(strategy='mean'))
    ]), numeric_features)
])

# Function to fill missing values using Linear Regression
def fill_missing(df, target_col, feature_cols):
    df_train = df[df[target_col].notnull()]
    df_test = df[df[target_col].isnull()]
    if df_test.empty:
        return df
    X_train = df_train[feature_cols]
    y_train = df_train[target_col]
    X_test = df_test[feature_cols]

    pipeline = Pipeline(steps=[
        ('preprocessor', preprocessor),
        ('regressor', LinearRegression())
    ])

    pipeline.fit(X_train, y_train)
    predictions = pipeline.predict(X_test)
    df.loc[df[target_col].isnull(), target_col] = predictions
    return df

# Step 1: Fill AvgRainfall(mm) first
rainfall_features = ['State', 'District', 'Crop', 'Season', 'SowingMonth', 'HarvestMonth', 'CropDuration(Days)']
df = fill_missing(df, 'AvgRainfall(mm)', rainfall_features)

# Step 2: Fill the remaining columns using rainfall as a feature
all_features = rainfall_features + ['AvgRainfall(mm)']
for col in ['AvgMinHumidity(%)', 'AvgMaxHumidity(%)', 'Min Temp Avg', 'Max Temp Avg']:
    df = fill_missing(df, col, all_features)
```

Features Preprocessing

Label Encoding

Delimiter: ,

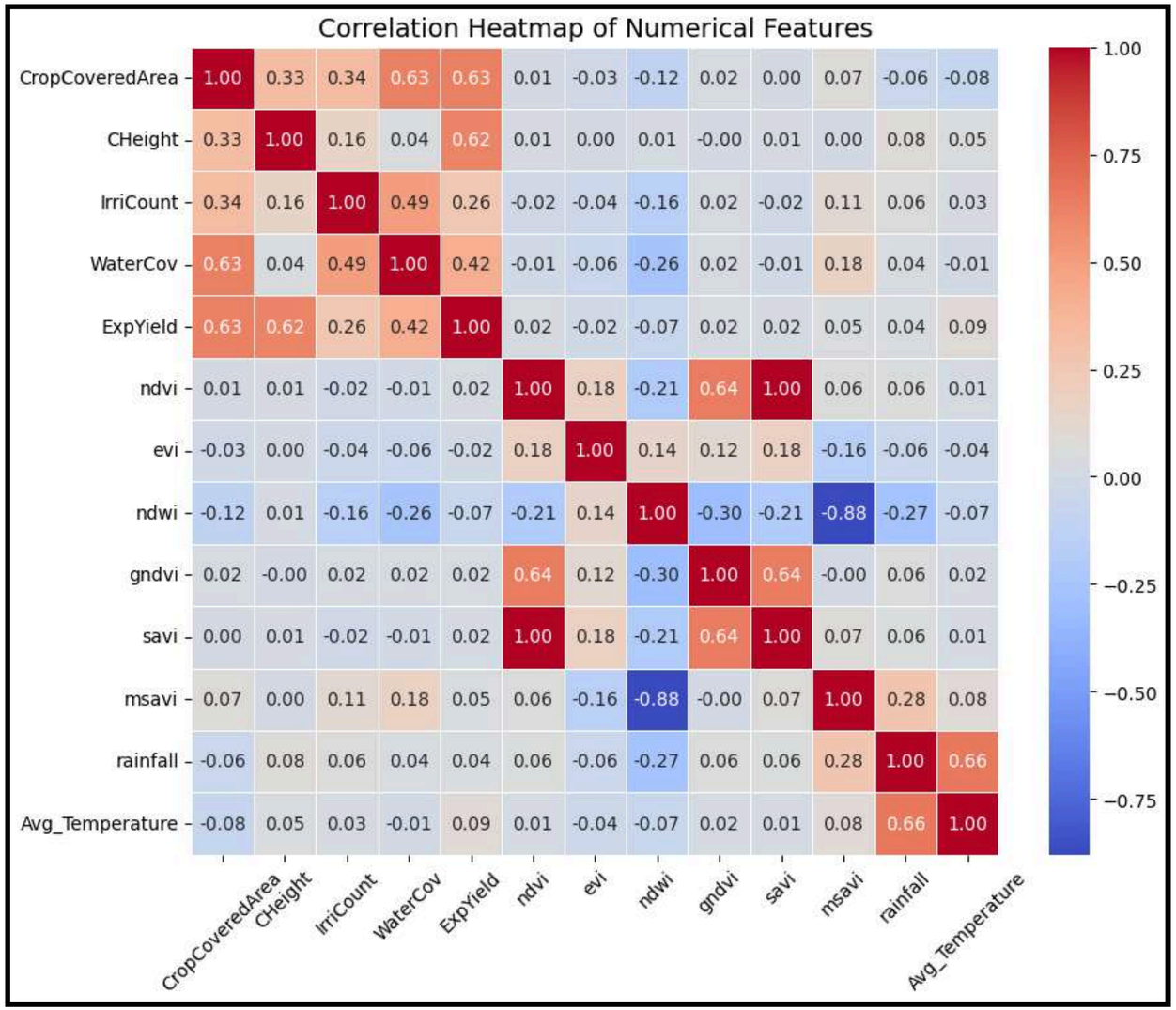
	CHeight	CNext	CLast	CTransp	IrriType	IrriSource	IrriCount	WaterCo
1	54	Pea	Lentil	Transplanting	Flood	Groundwater	4	8
2	58	Pea	Lentil	Transplanting	Flood	Canal	5	9
3	91	Pea	Lentil	Transplanting	Flood	Canal	3	9
4	52	Pea	Lentil	Transplanting	Flood	Canal	5	9
5	55	Pea	Lentil	Transplanting	Flood	Canal	5	9
6	51	Pea	Lentil	Transplanting	Flood	Groundwater	5	8
7	68	Lentil	Pea	Transplanting	Flood	Canal	3	9
8	72	Pea	Lentil	Transplanting	Flood	Canal	3	9
9	78	Pea	Lentil	Transplanting	Flood	Canal	3	8
10	53	Lentil	Pea	Transplanting	Flood	Groundwater	5	9
11	56	Lentil	Pea	Transplanting	Flood	Canal	4	9
12	59	Lentil	Pea	Transplanting	Flood	Canal	3	9
13	72	Pea	Lentil	Transplanting	Flood	Groundwater	6	8
14	51	Lentil	Pea	Transplanting	Flood	Groundwater	5	8
15	75	Lentil	Pea	Transplanting	Flood	Canal	6	8
16	62	Lentil	Pea	Transplanting	Flood	Groundwater	4	9
17	64	Lentil	Pea	Transplanting	Flood	Canal	6	9
18	99	[illegible]	[illegible]	Transplanting	Flood	Canal	6	8
19	94	Lentil	Pea	Transplanting	Flood	Canal	5	9
20	50	Pea	Lentil	Transplanting	Flood	Canal	4	8
21	61	Lentil	Pea	Transplanting	Flood	Canal	5	8
22	84	Lentil	Pea	Transplanting	Flood	Groundwater	3	9
23	60	Pea	Lentil	Transplanting	Flood	Canal	4	8

Delimiter: ,

	CHeight	CNext	CLast	CTransp	IrriType	IrriSource	IrriCount	WaterC
1	54	4	0	Transplanting	1	1	4	
2	58	4	0	Transplanting	1	0	5	
3	91	4	0	Transplanting	1	0	3	
4	52	4	0	Transplanting	1	0	5	
5	55	4	0	Transplanting	1	0	5	
6	51	4	0	Transplanting	1	1	5	
7	68	0	4	Transplanting	1	0	3	
8	72	4	0	Transplanting	1	0	3	
9	78	4	0	Transplanting	1	0	3	
10	53	0	4	Transplanting	1	1	5	
11	56	0	4	Transplanting	1	0	4	
12	59	0	4	Transplanting	1	0	3	
13	72	4	0	Transplanting	1	1	6	
14	51	0	4	Transplanting	1	1	5	
15	75	0	4	Transplanting	1	0	6	
16	62	0	4	Transplanting	1	1	4	
17	64	0	4	Transplanting	1	0	6	
18	99	4	0	Transplanting	1	0	6	
19	94	0	4	Transplanting	1	0	5	
20	50	4	0	Transplanting	1	0	4	
21	61	0	4	Transplanting	1	0	5	
22	84	0	4	Transplanting	1	1	3	
23	60	4	0	Transplanting	1	0	4	

Features Preprocessing

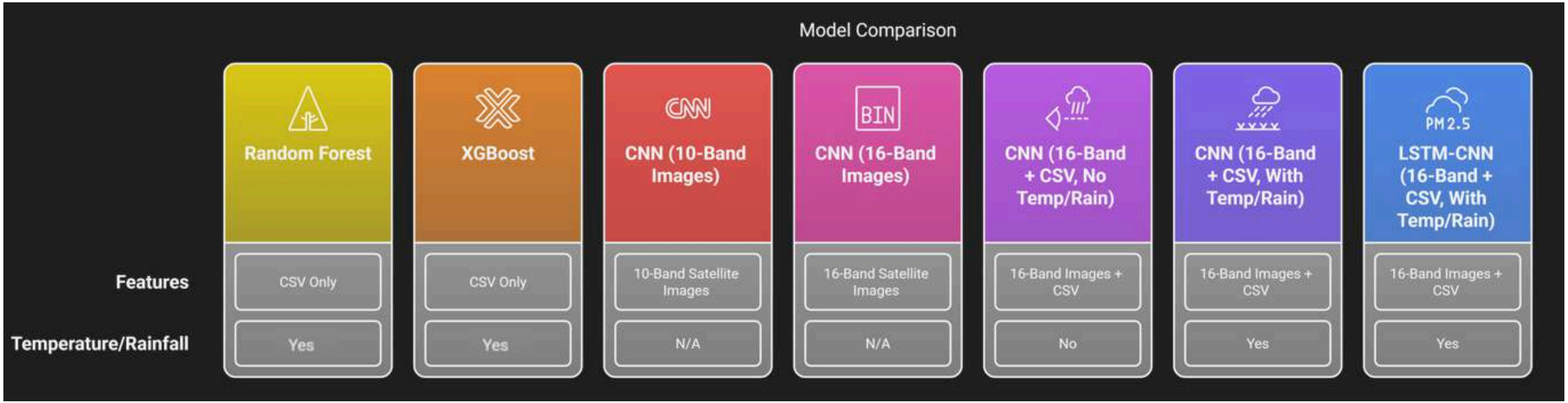
Correlation Heatmap



PROPOSED ML METHODOLOGY

1. Dataset: Labeled Sentinel-2 dataset with extracted vegetation indices (NDVI, NDWI, etc.).
2. Baseline Model: Train a Random Forest model for initial classification.
3. Advanced Models:
    a. Convolutional Neural Network (CNN) and Long-Short Term Memory (LSTM) to capture spatial and temporal patterns.
4. Enhancements:
    a. Hyperparameter tuning for optimization.
    b. Addition of more vegetation indices and texture-based features.
    c. Ensemble learning for improved accuracy.
5. Validation: Metrics like accuracy, precision, recall, and F1-score.
6. Final Goal: Develop a robust model for accurate crop health classification before deployment.

MODEL ARCHITECTURES



ARCHITECTURE 1
RF WITH ONLY CSV FEATURES

Why this model?

We began with Random Forest (RF) as a baseline ML approach due to its simplicity, robustness to overfitting, and interpretability.

How it works:

- RF creates multiple decision trees using bootstrapped datasets and aggregates their predictions (majority vote).
- It handles feature interactions and noisy data well.

Challenges faced:

- The CSV features lacked discriminatory power for stressed, diseased, and pest-affected classes.
- Severe class imbalance meant that the model performed well only for the healthy class.
- Limited interpretability of what features mattered.

Result:

Low recall and precision for minority classes → motivated shift to a more advanced model.

ARCHITECTURE 1
RF WITH ONLY CSV FEATURES

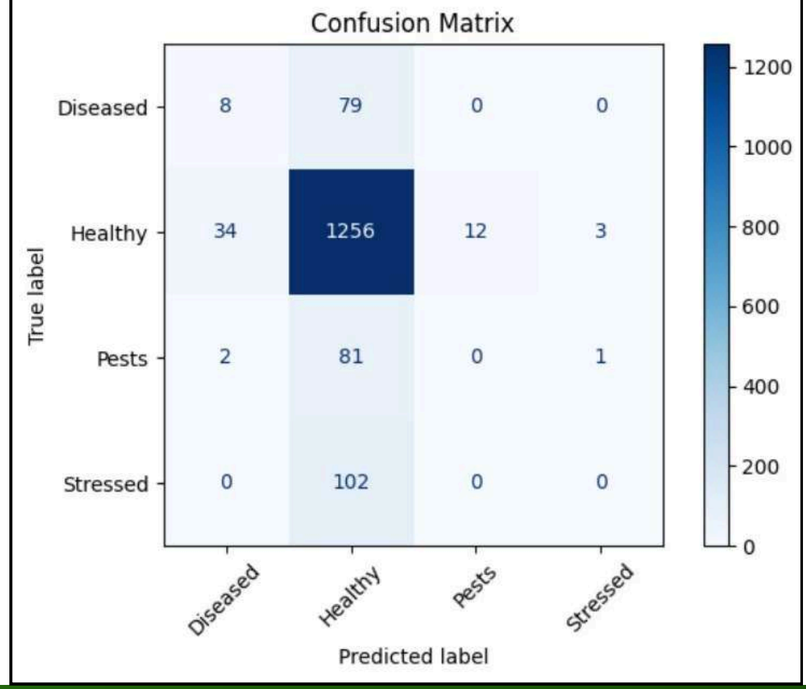
```
Accuracy: 0.8010139416983524

F1 Score (weighted): 0.7426227410238032

Classification Report (F1 per class):
              precision    recall  f1-score   support

    Diseased       0.18      0.09      0.12        87
     Healthy       0.83      0.96      0.89      1305
       Pests       0.00      0.00      0.00        84
    Stressed       0.00      0.00      0.00       102

    accuracy                           0.80      1578
   macro avg       0.25      0.26      0.25      1578
weighted avg       0.69      0.80      0.74      1578
```



ARCHITECTURE 2
XGBOOST WITH ONLY CSV FEATURES

Why this model?

We chose XGBoost for its:

- Boosting capability to reduce bias.
- Better handling of imbalanced classes and complex feature interactions.

How it works:

- Builds trees sequentially, each correcting errors of the previous.
- Uses gradient descent on a custom loss function.

Challenges faced:

- Despite theoretical improvements, performance was similar to RF.
- CSV data alone did not provide enough signal for distinguishing between crop health classes.

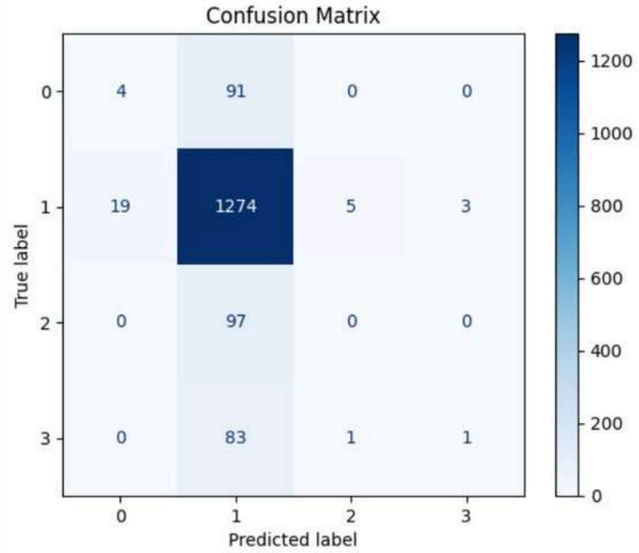
Conclusion:

No major gain → highlighted the limitation of necessary features->moved to Deep Learning.

ARCHITECTURE 2
XGBOOST WITH ONLY CSV FEATURES

```
Accuracy: 0.8105196451204055
F1 Score (weighted): 0.743425355237379

F1 Score for each class:
Class 0: 0.0678
Class 1: 0.8953
Class 2: 0.0000
Class 3: 0.0225
```




```
Classification Report:
              precision    recall  f1-score   support

         0.0       0.17      0.04      0.07        95
         1.0       0.82      0.98      0.90      1301
         2.0       0.00      0.00      0.00        97
         3.0       0.25      0.01      0.02        85

    accuracy                           0.81      1578
   macro avg       0.31      0.26      0.25      1578
weighted avg       0.70      0.81      0.74      1578
```

ARCHITECTURE 3
CNN WITH 10 BANDS
(ONLY SATELLITE IMAGES)

Why this model?

We collected Sentinel-2 tif images (10 bands) to capture spatial information beyond tabular metadata.

How it works:

- CNNs extract spatial features like textures, color variations, and patterns.
- Helpful in capturing signs of chlorosis, discoloration, or hotspots.

Challenges faced:

- Training was computationally expensive.
- Hard to interpret CNN filters and align them with agronomic knowledge.
- NDVI and vegetation-specific bands were missing.

Result:

Improved accuracy, especially for visual cues of stress, but lacked features like vegetation indices.

ARCHITECTURE 3
CNN WITH 10 BANDS
(ONLY SATELLITE IMAGES)

📊 Classification Report:

	precision	recall	f1-score	support
Diseased	0.30	0.81	0.44	32
Healthy	0.97	0.73	0.83	417
Pests	0.48	0.78	0.59	27
Stressed	0.41	0.92	0.56	24
accuracy			0.75	500
macro avg	0.54	0.81	0.61	500
weighted avg	0.87	0.75	0.78	500

ARCHITECTURE 4
CNN WITH 16 BANDS
(ONLY SATELLITE IMAGES)

Why this model?
To improve prediction, we added 6 vegetation index bands (like NDVI, EVI), expanding to 16-band imagery.

How it works:

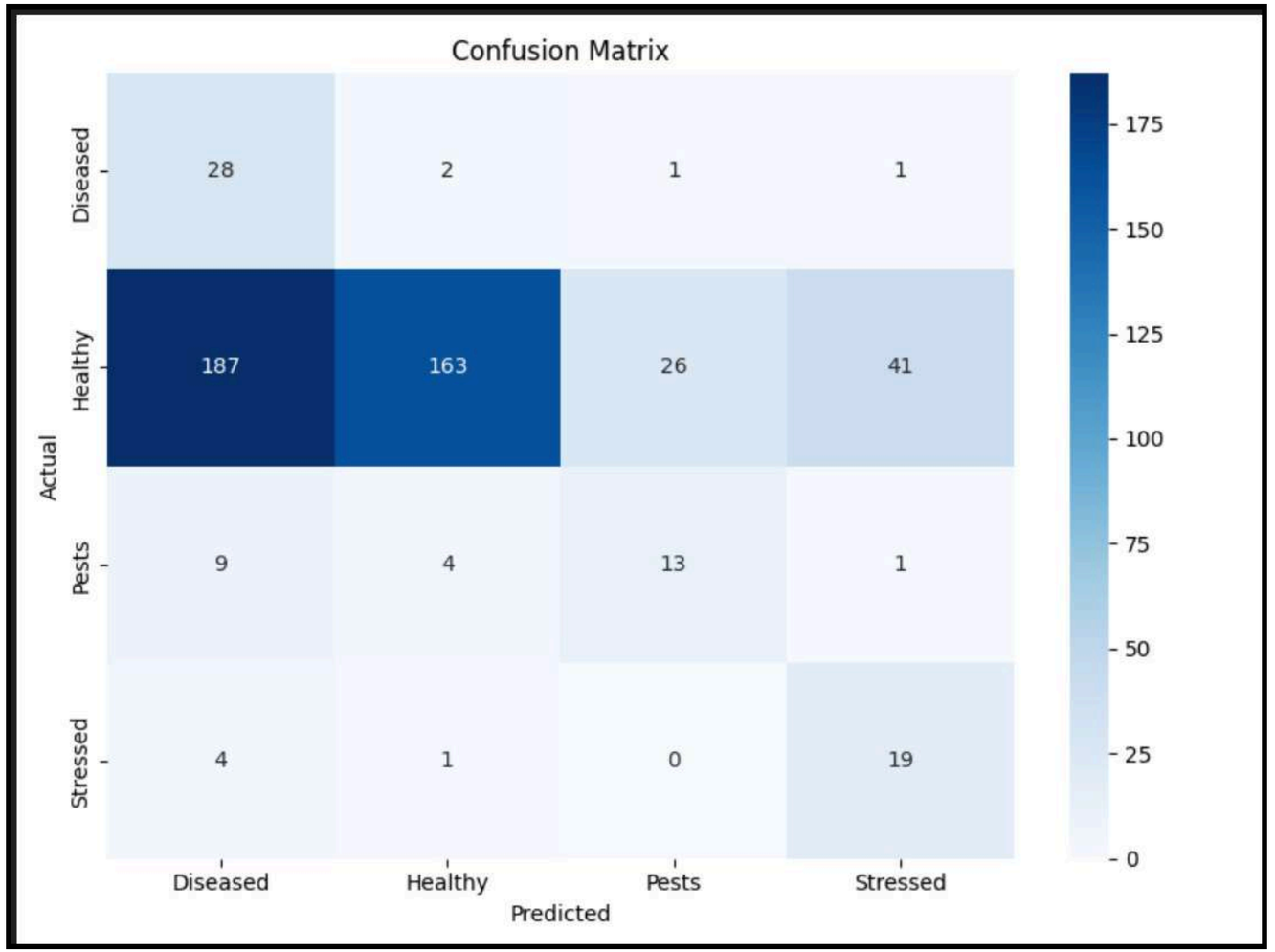
- CNN uses the extra bands to detect chlorophyll breakdown, water stress, and other spectral indicators.
- These bands directly correlate with crop health.

Challenges faced:

- Data preprocessing became heavier (band alignment, normalization).
- Model tuning required more epochs and memory.

Result:
Higher class separation for diseased crops, but wrong classification for healthy class.

ARCHITECTURE 4
CNN WITH 16 BANDS
(ONLY SATELLITE IMAGES)



ARCHITECTURE 5
CNN WITH 16 BANDS + CSV (NO TEMP/RAINFALL)

Why this model?
To combine spectral features with soil/environmental tabular data (e.g., Expected yield, Water covered Area).

How it works:

- Dual input model: CNN processes images; FC layers process CSV features.
- Fused features used for final prediction.

Challenges faced:

- Required custom data loader for multimodal inputs.
- Normalization mismatch between image and CSV data initially caused instability.

Result:
More context-aware predictions, but lacked weather-related temporal features.

ARCHITECTURE 5
CNN WITH 16 BANDS + CSV
(NO TEMP/RAINFALL)

📊 Classification Report:

	precision	recall	f1-score	support
Diseased	0.18	0.86	0.30	95
Healthy	0.96	0.68	0.80	1299
Pests	0.63	0.73	0.68	97
Stressed	0.73	0.82	0.78	84
accuracy			0.70	1575
macro avg	0.63	0.77	0.64	1575
weighted avg	0.88	0.70	0.76	1575

✅ Saved confusion matrix as 'confusion_matrix_dual_model_val.png'

ARCHITECTURE 6
CNN WITH 16 BANDS + CSV (WITH TEMP/RAINFALL)

Why this model?

Weather is a major factor in crop health. Hence, we added temperature, humidity and rainfall data.

How it works:

- Similar dual-stream architecture.
- Weather features improved understanding of drought or humidity-related stress.

Challenges faced:

- Sourcing and aligning local weather data for each image was difficult.
- Determining which features have the most impact required us to review prior research studies.
- Required time-synced data preprocessing.

Result:

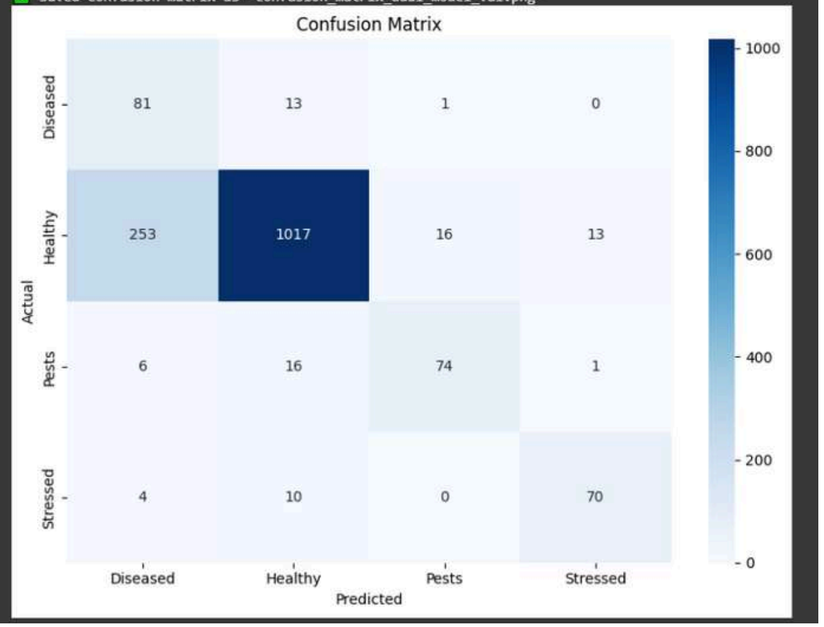
Significantly better performance for both Stressed-affected and pest-prone conditions.

ARCHITECTURE 6
CNN WITH 16 BANDS + CSV (WITH TEMP/RAINFALL)

📊 Classification Report:

	precision	recall	f1-score	support
Diseased	0.24	0.85	0.37	95
Healthy	0.96	0.78	0.86	1299
Pests	0.81	0.76	0.79	97
Stressed	0.83	0.83	0.83	84
accuracy			0.79	1575
macro avg	0.71	0.81	0.71	1575
weighted avg	0.90	0.79	0.83	1575

✅ Saved confusion matrix as `confusion_matrix_dual_model_val.png`



ARCHITECTURE 7
LSTM-CNN WITH CSV + WEATHER DATA

Why this model?

Crop health evolves over time. We added temporal dynamics using LSTM over sequences of NDVI and weather trends.

How it works:

- CNN extracts spatial features.
- LSTM models time-series patterns like NDVI decline or rainfall droughts.
- Final dense layer integrates all features.

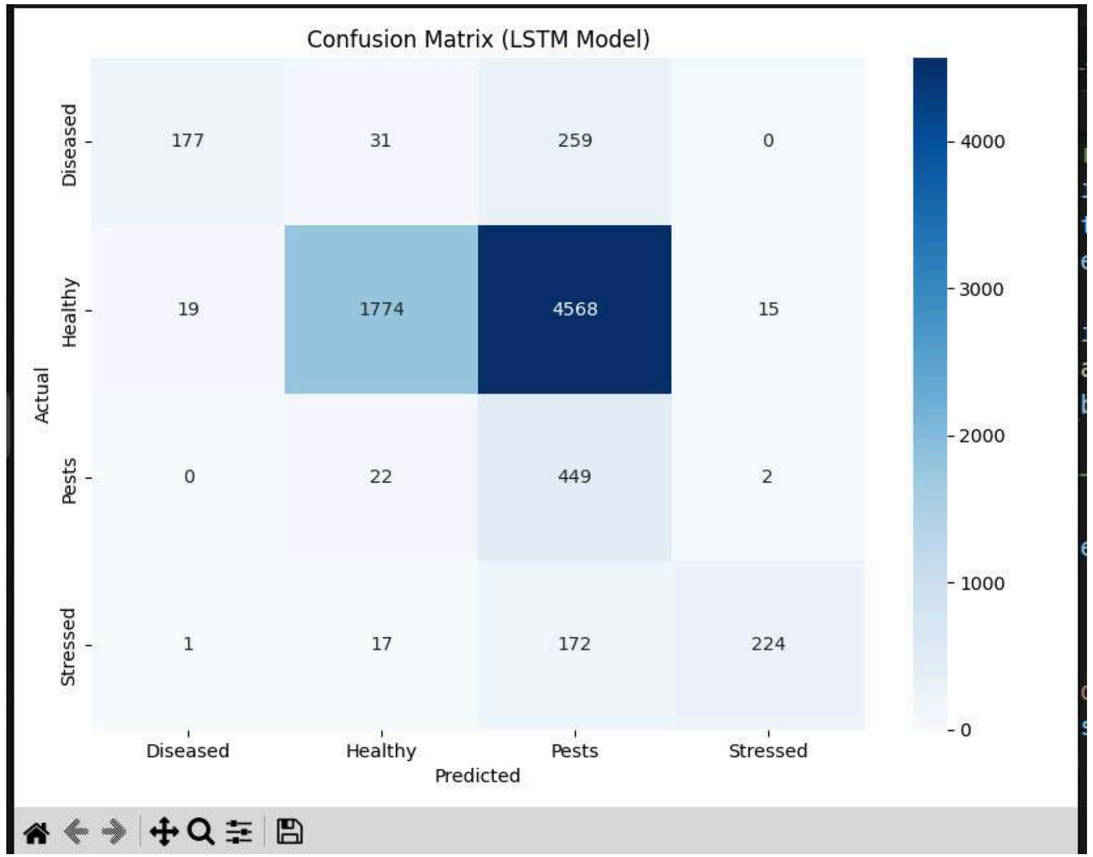
Challenges faced:

- Hard to collect enough time-series sequences for many fields.
- LSTM tuning and overfitting were initial issues.
- Class balancing across timeframes was complex.

Result:

Best performing model; accurately detected progressive stress, pest outbreaks, and hidden decay patterns.

ARCHITECTURE 7
LSTM-CNN WITH CSV + WEATHER DATA



PERFORMANCE METRICS & THEIR SIGNIFICANCE

1. Key Metrics Used

- **Accuracy:**Overall proportion of correctly classified samples.
  Not reliable for imbalanced datasets (Chicco, 2020).
- **Precision:** Ratio of true positives to total predicted positives — shows how many predicted "diseased" were actually diseased.
- **Recall:** Ratio of true positives to actual positives — shows how many diseased crops were detected correctly.
- **F1-Score:** Harmonic mean of precision and recall — balances false positives and false negatives. (Sasaki, 2007)
- **Macro Average**: Calculates the metric for each class independently and takes the unweighted mean — treats all classes equally.
- **Weighted Average**: Calculates metrics per class and averages them based on class frequency — better reflects performance on imbalanced data.

2. Why These Metrics?

- Multi-class imbalance: Healthy crops dominate the dataset.
- F1-score: Highlights performance on minority classes like stressed or pest-affected crops.
- Precision: Ensures fewer false alarms.
- Recall: Ensures real issues are not missed.
- Macro Average: Ensures all classes (even small ones) are equally evaluated.
- Weighted Average: Gives a realistic overall performance considering class distribution.

Sasaki, Y. (2007). The truth of the F-measure. Teach tutor mater, 1(5), 1-5.
Chicco, D., & Jurman, G. (2020). https://doi.org/10.1186/s12864-019-6413-7

PLAKSHA DEPLOYMENT POTENTIAL AND SCALING BARRIERS

Yes — Deployment Feasible in Plaksha's Sugarcane Fields or nearby fields

- Satellite-based monitoring can be implemented using Sentinel-2 imagery of Plaksha’s fields. First, we need to train it on sugarcane crop then it should work.
- Our model is already trained on crop health classification, enabling initial deployment.
- Can be adapted further using transfer learning or reinforcement learning as local data grows.

Challenges in Scaling Up

1. Regional Generalization

- Our dataset was Telangana-specific, and Plaksha may have different climate patterns (temperature, rainfall, soil).
- Model may misclassify due to unseen regional variations.

2. Crop-Specific Bias

- Trained on 7 crops.
- May not generalize to Plaksha’s sugarcane or other local crops, thus there is a need for crop-specific data.

3. Data Diversity & Quantity

- Scaling requires continuous updates and retraining with newer images and ground truth labels.

4. **Need for More Data to Generalize**

- To scale effectively, we need diverse datasets from multiple regions and seasons.
- Incorporating edge cases and anomalies will improve model reliability in real-world deployment.

KEY CHALLENGES

1. Overfitting & Model Generalization

- Deep learning models started overfitting early despite regularization.
- Complex CNN architectures didn’t scale well on limited data.

2. Dataset Limitations

- Limited diversity: Dataset was Telangana-specific — lacked generalization across regions.
- Data scarcity: Each time-series input required ~5 satellite images per field — difficult to compile.
- Imbalanced classes: "Healthy" dominated; other classes like "stressed" or "pest-affected" were underrepresented.

3. Computational Constraints

- High GPU demand for training multimodal and sequential models (CNN + LSTM).
- Only one team member had GPU access — bottlenecked parallel experiments.

4. Hyperparameter Tuning

- Balancing layer complexity, dropout, learning rate, batch size was time-consuming.
- Automated tuning tools were limited by hardware and time.

5. Complex Conv layers

- Overfitting and val loss increased

HOW WE OVERCAME THE CHALLENGES

1. Tackling Overfitting

- Implemented oversampling and undersampling, data augmentation, dropout, and early stopping.
- Reduced CNN complexity instead of increasing — shifted focus to feature quality.
- Applied class-balanced sampling and oversampling techniques.

2. Improving Data Pipeline

- Spent time understanding Sentinel-2 image bands and indices (NDVI, EVI).
- Built a custom time-series image generator for efficient batch loading.
- Prioritized temporal consistency over raw volume.

3. Leveraging Collective Effort

- Took help from student TAs, professors from the robotics lab, and domain experts.
- Studied over 50+ research papers on remote sensing and crop classification.

4. Efficient Collaboration & Learning

- Divided tasks: data preprocessing, model building, training, and visualization.
- Weekly verbal reports regarding progress and then updating the TAs.
- Weekly sync-ups kept everyone aligned despite hardware disparity.

FUTURE WORK & NEXT STEPS

1. Temporal Climate Focus

- Use temperature and rainfall data only during critical growth stages (e.g., blooming, harvesting) To better capture heat stress impact on crop health and yield.

2. Expand Crop Diversity

- Extend model training to include major crops grown across India. Enhances generalizability across different agricultural zones.

3. Model Optimization

- Perform hyperparameter tuning on the LSTM-CNN hybrid model

Aim: Improve prediction accuracy, especially for minority classes.

4. Explore Advanced Architectures

- Experiment with Transformers and Foundation Models. Leverage their capability to model long-term dependencies and complex feature relationships.

THANK YOU