Industrial Internship Report on

”Prediction of Agriculture Crop Production in India”

Prepared by

Rathore Rikendra Dolatsingh

Executive Summary

This report provides details of the Industrial Internship provided by upskill Campus and The IoT Academy in collaboration with Industrial Partner UniConverge Technologies Pvt Ltd (UCT).

This internship was focused on a project/problem statement provided by UCT. We had to finish the project including the report in 6 weeks’ time.

My Project was predicting Production of Agriculture Crop in India, included a robust data ingestion pipeline, exploratory data analysis, a leak-free forecasting model, and a Streamlit application for interactive predictions, backtesting, and batch forecasting. The project was completed within the internship timeline and packaged with documentation and reproducible code.

My project built a practical forecasting workflow:

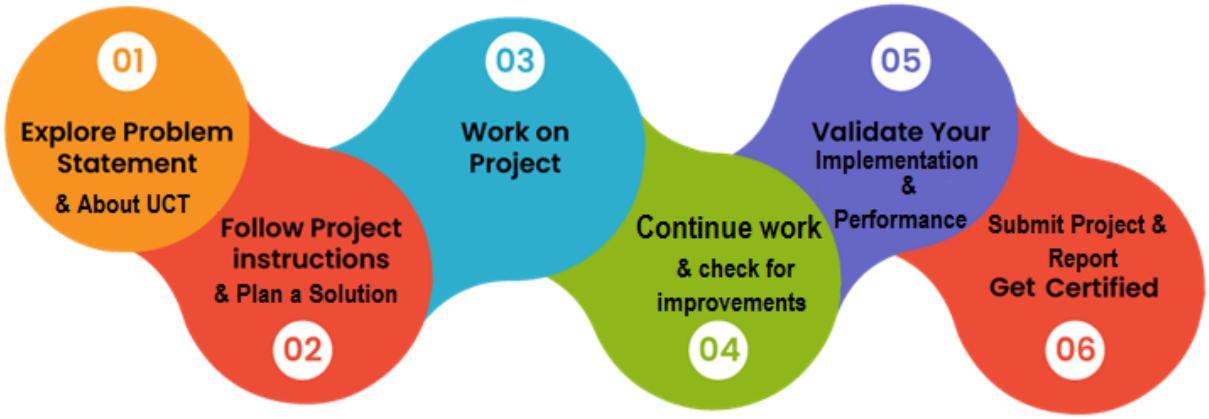
- Consolidated five heterogeneous CSV/Excel files (wide fiscal-year tables, “Particulars” series, mixed units) into a single clean dataset.
- Engineered forecast-safe features (only prior-year information), trained a RandomForest model in a scikit-learn pipeline, and benchmarked against a naive baseline.
- Delivered a Streamlit app with three tabs: Single prediction, Evaluate (backtest), and Batch forecast.

This internship gave me a very good opportunity to get exposure to Industrial problems and design/implement solution for that. It was an overall great experience to have this internship.

TABLE OF CONTENTS

1 Preface

This report summarizes my internship work from project scoping to delivery. The internship provided practical industry exposure to data engineering, machine learning, and application development. The problem addressed—forecasting crop production for Indian agriculture—has social and economic significance, enabling better planning and policy insights.



I am grateful to:

- upskill Campus and The IoT Academy for organizing the program and mentorship,
- UniConverge Technologies Pvt. Ltd. for the problem statement and industrial context,
- My mentors and peers for their guidance on best practices in data/ML and deployment.

Overall, the internship strengthened my fundamentals in ETL, time-aware modeling, and streamlit-based productization, and improved my communication and problem-solving skills.

2 Introduction

2.1 About UniConverge Technologies Pvt Ltd

A company established in 2013 and working in Digital Transformation domain and providing Industrial solutions with prime focus on sustainability and RoI.

For developing its products and solutions it is leveraging various **Cutting Edge Technologies e.g. Internet of Things (IoT), Cyber Security, Cloud computing (AWS, Azure), Machine Learning, Communication Technologies (4G/5G/LoRaWAN), Java Full Stack, Python, Front end** etc.

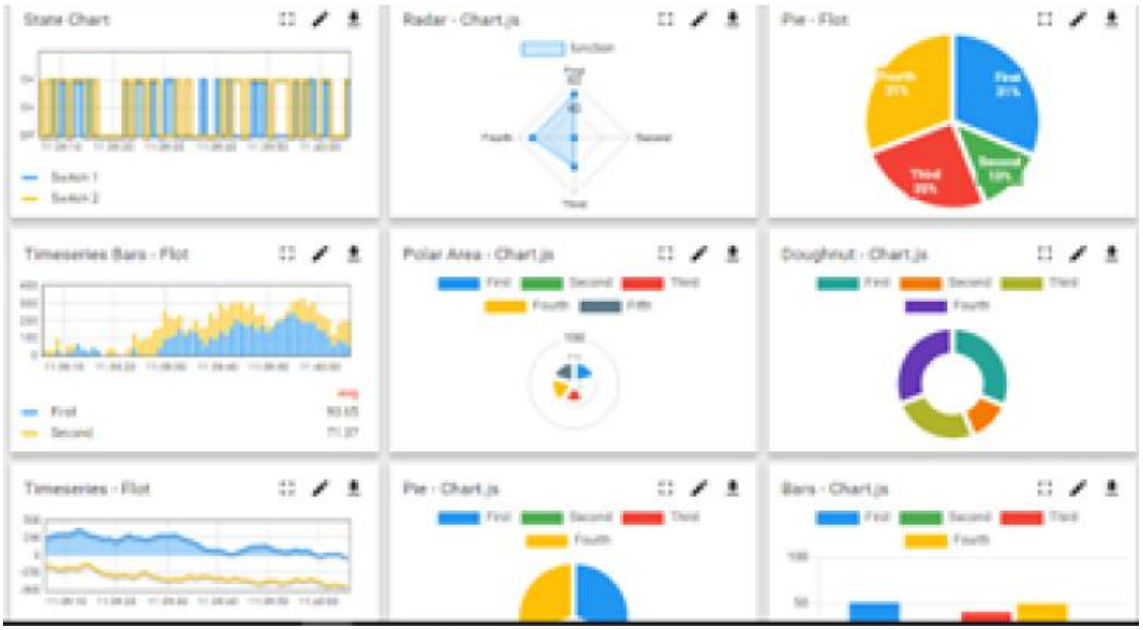
i. UCT IoT Platform (UCT Insight)

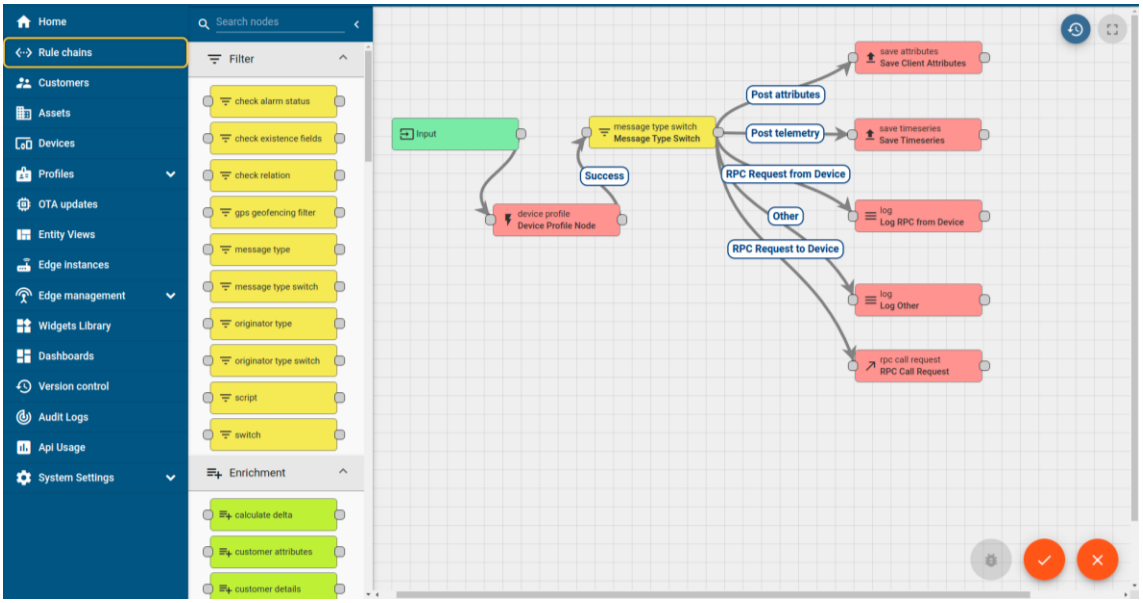
UCT Insight is an IOT platform designed for quick deployment of IOT applications on the same time providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- It enables device connectivity via industry standard IoT protocols - MQTT, CoAP, HTTP, Modbus TCP, OPC UA
- It supports both cloud and on-premises deployments.

It has features to

- Build Your own dashboard
- Analytics and Reporting
- Alert and Notification
- Integration with third party application(Power BI, SAP, ERP)
- Rule Engine





ii. Smart Factory Platform (FACTORY WATCH)

Factory watch is a platform for smart factory needs.

It provides Users/ Factory

- with a scalable solution for their Production and asset monitoring
- OEE and predictive maintenance solution scaling up to digital twin for your assets.
- to unleased the true potential of the data that their machines are generating and helps to identify the KPIs and also improve them.
- A modular architecture that allows users to choose the service that they what to start and then can scale to more complex solutions as per their demands.

Its unique SaaS model helps users to save time, cost and money.

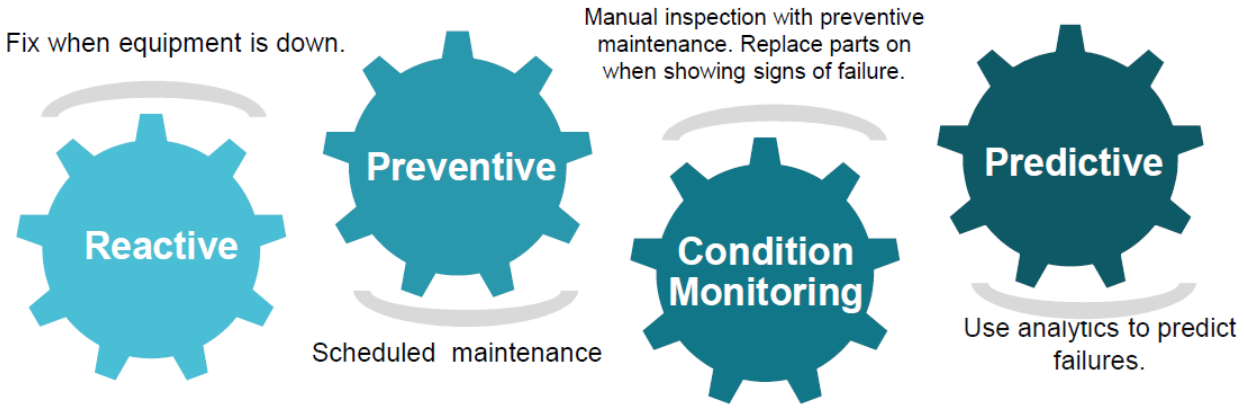
Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i

iii. LoRaWAN based Solution

UCT is one of the early adopters of LoRAWAN teschnology and providing solution in Agritech, Smart cities, Industrial Monitoring, Smart Street Light, Smart Water/ Gas/ Electricity metering solutions etc.

iv. Predictive Maintenance

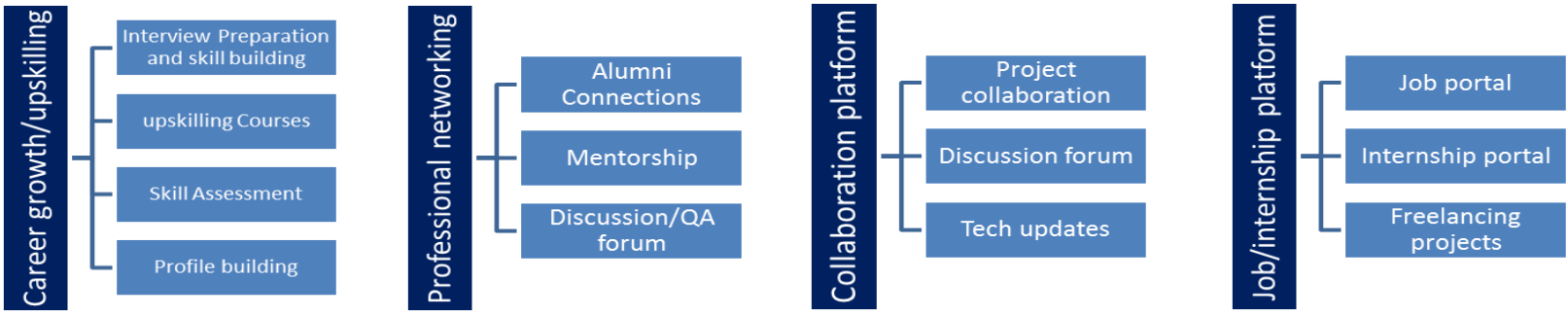
UCT is providing Industrial Machine health monitoring and Predictive maintenance solution leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life time of various Machines used in production process.



2.2 About upskill Campus (USC)

upskill Campus along with The IoT Academy and in association with Uniconverge technologies has facilitated the smooth execution of the complete internship process.

USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.



2.3 The IoT Academy

The IoT academy is EdTech Division of UCT that is running long executive certification programs in collaboration with EICT Academy, IITK, IITR and IITG in multiple domains.

2.4 Objectives of this Internship program

The objective for this internship program was to

- ☛ get practical experience of working in the industry.
- ☛ to solve real world problems.
- ☛ to have improved job prospects.
- ☛ to have Improved understanding of our field and its applications.
- ☛ to have Personal growth like better communication and problem solving.

2.5 Reference

[1] data.gov.in – Agriculture datasets (Government of India Open Data)
[2] pandas documentation – https://pandas.pydata.org
[3] scikit-learn documentation – https://scikit-learn.org
[4] Streamlit documentation – https://docs.streamlit.io
[5] NumPy documentation – https://numpy.org/doc

2.6 Glossary

Terms	Acronym
ETL	Extract, Transform, Load
EDA	Exploratory Data Analysis
FY	Fiscal Year (e.g., 2006–07 → end year 2007)
OHE	One-Hot Encoding
Baseline (naive)	predict this year = last year (per crop)

3 Problem Statement

In the assigned problem statement

Build a forecasting model to predict annual agriculture crop production (in Tons) for India using historical data (2001–2014, with effective years used from available files). The raw data arrives in multiple heterogeneous formats—wide fiscal-year tables, “Particulars” series with varying units, and mixed schemas—requiring a robust ingestion layer. The solution should be time-aware (no data leakage), benchmarked against a naive baseline, and exposed as a simple application stakeholders can use.

4 Existing and Proposed solution

Existing solutions:

- Static reports and spreadsheets often provide historical values without interactive forecasts.
- Academic models may not handle messy real-world data ingestion or provide user-facing tools.

Limitations addressed:

- Heterogeneous raw files and units cause friction; forecasting workflows are non-reproducible.
- Lack of an easy interface for quick single predictions, backtesting, and batch forecasts.

Proposed solution:

- A complete pipeline: ingestion → EDA → forecast-safe feature engineering → time-aware training and evaluation → Streamlit app.
- Ingestion that standardizes schemas, converts fiscal-year columns (2006–07 → 2007), normalizes units to Tons, and melts wide tables to a long format.
- Model trained with only prior-year information (lags), avoiding leakage.
- App with three workflows: Single prediction, Evaluate (backtest), Batch forecast.

Value addition:

- Reproducible, auditable data pipeline with provenance.
- Strong baseline and model comparison with published metrics.
- Practical app for both exploration and planning.

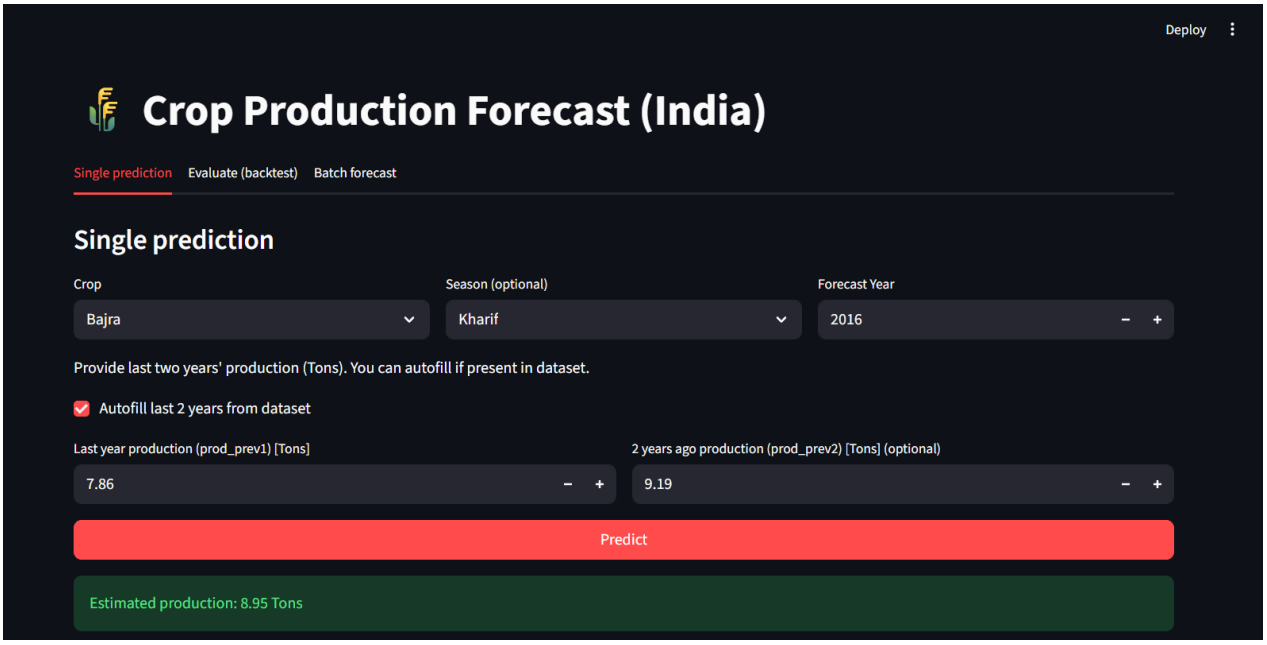
4.1 Code submission (Github link)

Repository: https://github.com/Rikendra-Rathore/upskillcampus.git

Report submission (Github link):

{https://github.com/Rikendra-Rathore/upskillcampus.git}

5 Proposed Design/ Model



	crop	season	year	prod_prev1	prod_prev2	prediction_tons
38	Jowar	Kharif	2014	2.84	3.293	2.8556
39	Jowar	Rabi	2014	2.44	2.6862	3.1178
40	Jute	None	2014	None	None	4.418
41	Jute & Mesta	None	2014	None	None	4.418
42	Karnataka	None	2014	10.93	12.1	12.61
43	Kashmir	None	2014	545.6	544.7	113.5775
44	Kerala	None	2014	530	570	114.7575
45	Linseed	Rabi	2014	0.149	0.1525	0.1517
46	Maharashtra	None	2014	10.69	12.54	12.8384
47	Maize	Kharif	2014	16.19	16.4863	15.7403

5.1 High Level Diagram (if applicable)

Raw Files (CSV/Excel: data/raw)
→ Ingestion (src/ingest.py)
- Standardize headers
- Reshape wide FY columns
- Unit normalization (→ Tons)
- Dedup + provenance
→ Combined Dataset (data/interim/agri_combined.csv)
→ Notebook (notebooks/01_eda.ipynb)
- EDA
- Feature engineering (lags)
- Time-aware split
- Baseline + Model training
- Save model + meta (models/)
→ Streamlit App (app/app.py)
- Single prediction
- Evaluate (backtest)
- Batch forecast

Figure 1: HIGH LEVEL DIAGRAM OF THE SYSTEM

5.2 Low Level Diagram (if applicable)

- Ingestion
  - Detect files (*.csv, *.xls, *.xlsx)
  - Canonicalize column names (snake_case)
  - Map synonyms (state_name→state, crop_year→year, etc.)
  - Reshape:
    - Detect production_YYYY_YY, area_YYYY_YY, yield_YYYY_YY → melt to long
    - Fallback: “Particulars” + generic FY columns (e.g., 2004_05) → production series
  - Units:
    - “Thousand Tonnes” → ×1000 Tons
    - “Quintals” → ÷10 Tons
  - Save CSV with source_file/source_sheet columns
- Modeling
  - Features (forecast-safe):
    - prod_prev1, prod_prev2, prod_ma2 (lagged 2-yr mean), prod_delta
    - Optional: area/yield lags and deltas when present
    - Categorical: crop, season
    - Numeric: year
  - Pipeline:
    - SimpleImputer (median for numeric, most_frequent for categorical)
    - OneHotEncoder (handle_unknown=ignore)
    - RandomForestRegressor (n_estimators≈600, random_state=42)
  - Baseline: last-year = this-year

5.3 Interfaces (if applicable)

Update with Block Diagrams, Data flow, protocols, FLOW Charts, State Machines, Memory Buffer Management.

- File paths:
    - Input: data/raw/*.csv, *.xls, *.xlsx
    - Output dataset: data/interim/agri_combined.csv
    - Model artifacts: models/production_predictor.joblib, models/meta.json
- CLI:
    - Ingest: python src/ingest.py
    - App: python -m streamlit run app/app.py
- App tabs:
    - Single prediction (manual/autofill prior years)
    - Evaluate (crop/season range backtest; metrics + charts + CSV)
    - Batch forecast (all crops for a target year; CSV)

6 Performance Test

Constraints and design choices:

- Data heterogeneity (schemas, FY formats, units) → resilient ingestion with regex parsing and normalization.
- Limited years per crop in some files → designed features using only prior-year data (prevents leakage and supports small-sample settings).
- Missing values (NaNs) → imputation inside the pipeline for robust training/inference.
- Reproducibility → fixed randomstate, version pinning, provenance fields.

6.1 Test Plan/ Test Cases

- Ingestion
    - TP-ING-01: Reads all files; creates data/interim/agri_combined.csv; non-zero rows.
    - TP-ING-02: FY mapping (2006–07 → 2007) verified on sample.
    - TP-ING-03: Unit normalization: Thousand Tonnes → Tons; Quintals → Tons.
    - TP-ING-04: Dedupe and provenance columns present.
- Modeling
    - TP-ML-01: Feature lags computed correctly (no future leakage).
    - TP-ML-02: Time-aware split by year (train ≤ Y−2, valid = Y−1, test = Y).
    - TP-ML-03: Baseline computed and logged.
    - TP-ML-04: Pipeline handles NaNs via imputers; training succeeds.
- App
    - TP-APP-01: Single prediction works with autofill and manual inputs.
    - TP-APP-02: Evaluate tab generates metrics, chart, and CSV download.
    - TP-APP-03: Batch forecast runs for all crops; exports CSV.

- TP-APP-04: Paths (debug) indicate MODEL/META/DATA found.

6.2 Test Procedure

- Run python src/ingest.py; check logs and open the combined CSV to validate schema and units.
- Execute notebooks/01_eda.ipynb end-to-end; record baseline and model metrics; save model/meta.
- Launch app with python -m streamlit run app/app.py; validate each tab's workflow with 2–3 crops and multiple years.

6.3 Performance Outcome

Dataset: consolidated long-format crop-year table; units normalized to Tons.

- Baseline (last-year):
  - Strong (as expected for short series), used as a reference.
- RandomForest Pipeline (features: year, prod_prev1, prod_prev2, prod_ma2, prod_delta, crop, season):
  - Validation: MAE ≈ 1.49, RMSE ≈ 3.25, R^2 ≈ 0.991
  - Test: MAE ≈ 1.93, RMSE ≈ 3.61, R^2 ≈ 0.989
- App:
  - Single prediction, Evaluate (backtest), and Batch forecast functioning with CSV downloads and debug path checks.

7 My learnings

- Data engineering: Building resilient ingestion for messy real-world data (wide→long, unit normalization, FY parsing, provenance).
- Time-aware modeling: Designing leak-free features (lags) and splits; interpreting MAE/RMSE/R^2/MAPE.
- ML engineering: scikit-learn pipelines (imputers + encoders + estimator), version pinning, and artifact management.
- App development: Streamlit UI/UX, caching, and structuring tabs for practical stakeholder use.
- Reproducibility and documentation: Clear repo structure, README, and runbook for smooth onboarding and evaluation.

8 Future work scope

- Add state-level granularity and external drivers (rainfall, temperature, irrigation coverage, fertilizer consumption) to improve prediction quality.
- Expand algorithms: CatBoost/XGBoost, hyperparameter tuning, and uncertainty estimation (prediction intervals).
- Comprehensive walk-forward backtesting across all crops and years; add MAPE alerts where data is sparse.
- Model explainability (SHAP feature importance) and scenario analysis (what-if changes in area or yield).
- Cloud deployment (Streamlit Community Cloud) with data-governance-friendly samples or on-prem option.