**Crop Type Classification – Notebook Walkthrough**

*A clear, step‑by‑step explanation of the final training notebook for project partners.*

## 1) What this model does

This model predicts the crop type for each field by learning patterns from satellite signals (Sentinel‑1 radar, Sentinel‑2 vegetation indices) and management/environmental variables (e.g., irrigation, seasonality). We train an XGBoost classifier, select the most informative features, and evaluate performance with grouped cross‑validation and an independent test year.

## 2) Workflow at a glance

0.1 Import libraries  
0.2 Set up parameters and file paths  
0.3 Load, clean, and merge datasets  
0.4 Train initial XGBoost to estimate feature importance  
0.5 Select top features and retrain  
0.6 Apply Forward Feature Selection (FFS) with Group K‑Fold CV  
0.7 Plot cumulative accuracy improvement and pick the elbow point  
0.8 Train the final model using the best features and evaluate on 2019 (held‑out)  
0.9 Visualize final feature importance  
0.10 Build and interpret two confusion matrices (CV and true test)  
0.11 Save model, features, and a summary file (accuracy + feature details)

## 0.1 Importing Required Libraries

We import the Python packages used throughout the notebook:  
• pandas, numpy – data cleaning and manipulation  
• xgboost, scikit‑learn, mlxtend – model training and feature selection  
• matplotlib, seaborn – charts  
• joblib, os, time – saving models and tracking runtime

## 0.2 Setup and Define Model Parameters

We centralize all settings so runs are reproducible and easy to change:  
• Input/output folders and filenames  
• XGBoost hyperparameters (number of trees, depth, learning rate, etc.)  
• Feature‑selection limits and early‑stopping threshold  
• Parallel processing (CPU cores)

## 0.3 Loading and Preprocessing the Dataset

We load two datasets (All, SB25r), keep only their common columns, and merge them into a single table. Columns are reordered to place identifiers first (FIELDID, Crop\_type, Crop\_num, Year). We then inspect class/Year distributions and visualize weekly average NDVI per crop to understand seasonal patterns.

## 0.4 Train the Initial XGBoost Model

We train an initial XGBoost model on all features to compute feature‑importance scores. Sample weighting is applied so smaller classes (e.g., Sorghum) are not overshadowed by larger ones (e.g., Pasture).

## 0.5 Feature Importance and Selection

We sort features by importance, keep the top N predictors, and retrain XGBoost on this reduced set. This focuses the next step (FFS) on the strongest candidates and reduces noise.

## 0.6 Forward Feature Selection (FFS)

We apply Sequential Forward Feature Selection to identify the best combination of predictors. Features are added one by one; a feature is kept only if it improves cross‑validated accuracy. The process stops when further gains are negligible (early‑stopping threshold).

Understanding K‑Fold Cross‑Validation:  
• The dataset is split into K equal parts (folds).  
• Train on K−1 folds, test on the remaining fold.  
• Repeat K times so every fold serves once as the test set.  
• Average scores across folds for a robust estimate on unseen data.

Why Group K‑Fold here?  
Multiple records come from the same field/year (e.g., observations of one parcel). Group K‑Fold keeps all records from the same group together: either entirely in train or in test, never split. This prevents data leakage and yields a fair estimate of performance on new fields/seasons.

## 0.7 Plotting Cumulative Accuracy Improvement

We plot cross‑validated accuracy at each FFS step. The “elbow point” shows when adding features stops improving results meaningfully. That point defines the final set (about 17–18 features in this run).

## 0.8 Train the Final XGBoost Model

We train the final model using the selected features. Steps include an 80/20 stratified split, class weights, and early stopping. Crucially, we evaluate on a held‑out 2019 test year never used in training. We compute precision, recall, F1, and overall accuracy.

## 0.9 Feature Importance of Final Model

We chart and save final feature importances to show which variables influenced decisions most (e.g., NDVI dynamics, week‑of‑year, irrigation, Sentinel‑1 radar metrics). This adds transparency.

## 0.10 Model Evaluation and Confusion Matrices

We present two complementary confusion matrices:  
1) Cross‑Validation (Group K‑Fold) – combines predictions from all folds to show overall consistency.  
2) Independent Test Year (2019) – shows real out‑of‑sample performance on unseen data. This second matrix is the most relevant for deployment decisions.

How to read them:  
• Diagonal = correct classifications; off‑diagonals = confusions between crops.  
• We show both raw counts and normalized percentages.  
• Figure titles include accuracy and macro F1 for quick comparison.

## Understanding Precision, Recall, F1‑score, and Support

• Precision → How often the model’s predictions are right (few false positives).  
• Recall → How many of the real cases the model actually finds (few missed cases).  
• F1‑score → The balance between precision and recall (high only if both are high).  
• Support → How many samples belong to that crop class (used to weight averages).

## 0.11 Save the Final Model and Summary Information

We export key artifacts so the model can be reused without retraining:  
• Trained model (.joblib)  
• Selected features (.json)  
• Confusion matrix image and CSV  
• Evaluation summary (.json) containing:  
 – Final\_Accuracy (overall share of correct predictions)  
 – Weighted\_F1 (balanced precision–recall, weighted by class size)  
 – Selected\_Features and Num\_Selected\_Features  
 – Paths to the saved model, feature list, and test‑set reports

## Takeaways

• The model achieves strong, balanced performance across major crops and generalizes well to a new year.  
• Group K‑Fold avoids leakage and yields realistic estimates for deployment.  
• Forward Feature Selection keeps the model compact while preserving accuracy.