Project 2: Dry Bean Classification — SVM (Linear/Poly/RBF) vs Decision Tree

Machine Learning

Due: Oct 15

Overview

In this project, you will use the **Dry Bean** dataset to compare three Support Vector Machine (SVM) kernels—**linear**, **polynomial**, and **RBF**—against a **Decision Tree** classifier. Your goal is to build a reproducible experiment, evaluate each model with appropriate metrics, and present concise results.

Requirements

1. **Data Preparation**
   - Load the Dry Bean dataset.
   - Perform a stratified train/test split (e.g., 80/20).
   - Apply scaling to features for SVM models using the training set only.
   - Do not scale data for the Decision Tree.
2. **Models to Train**
   - **SVM (Linear)**
   - **SVM (Polynomial)** with degree tuning.
   - **SVM (RBF)** with parameter tuning.
   - **Decision Tree** with depth and split parameter tuning.

   Use cross-validation on the training set for hyperparameter selection. Report final performance on the test set only.
3. **Evaluation**
   - Report **Macro-F1** as the primary metric.
   - Also include **Accuracy**, Precision, Recall, and per-class results.
   - Provide a **confusion matrix** for each model.
4. **Summary Sheet (1–2 pages)**

- A **results table** comparing all four models.
- At least one **plot** (e.g., bar chart of Macro-F1 or confusion matrix heatmaps).
- A short **takeaway paragraph** (5–8 sentences) discussing which model performed best, why that may be, and any evidence of overfitting or underfitting.

5. **Code Submission**

- Submit clean, well-documented `.py` files only.
- Code should be reproducible: fix random seeds and include any necessary run instructions.

Result Table Template

Model	**Macro-F1 (Test)**	**Accuracy (Test)**
SVM (Linear)	___	___
SVM (Poly, degree=__)	___	___
SVM (RBF)	___	___
Decision Tree	___	___

Submission

Upload a single `.zip` file that includes:

- `src/` directory with Python code.
- `summary.pdf` (1–2 page summary sheet).
- A short `README` with instructions to run your code.

Notes

- Fix random seeds (`random_state=42`) for reproducibility.
- Use stratified splits to handle class imbalance.
- Prefer **Macro-F1** for evaluation since the dataset is imbalanced.
- Keep the summary concise: this is not a full paper.