

Assignment No 1

```
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, r2_score
import joblib
import numpy as np

data = pd.read_csv("/content/RS-A1_yield.csv")

feature_cols = ['Area', 'Item', 'Year', 'average_rain_fall_mm_per_year',
'pesticides_tonnes', 'avg_temp']
target_col = 'hg/ha_yield'

X = data[feature_cols]
y = data[target_col]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

cat_features = ['Area', 'Item']
num_features = [c for c in feature_cols if c not in cat_features]

preprocessor = ColumnTransformer(
    transformers=[
        ('cat', OneHotEncoder(handle_unknown='ignore', sparse=False), cat_features)
    ],
    remainder='passthrough'
)

pipeline = Pipeline([
    ('pre', preprocessor),
    ('rf', RandomForestRegressor(n_estimators=200, random_state=42, n_jobs=-1))
])

pipeline.fit(X_train, y_train)

preds = pipeline.predict(X_test)
mae = mean_absolute_error(y_test, preds)
r2 = r2_score(y_test, preds)
print(f"MAE: {mae:.2f}")
```

```

print(f"R^2: {r2:.3f}")

cv_scores = cross_val_score(pipeline, X, y, cv=5, scoring='neg_mean_absolute_error',
n_jobs=-1)
cv_mae = -np.mean(cv_scores)
print(f"5-fold CV MAE: {cv_mae:.2f}")

rf = pipeline.named_steps['rf']
ohe = pipeline.named_steps['pre'].named_transformers_['cat']
try:
    cat_names = ohe.get_feature_names_out(cat_features).tolist()
except:
    cat_names = []
for i, feat in enumerate(cat_features):
    cats = ohe.categories_[i]
    cat_names += [f"{feat}_{c}" for c in cats]

feature_names = cat_names + num_features
importances = rf.feature_importances_
feat_imp = sorted(zip(feature_names, importances), key=lambda x: x[1], reverse=True)
print("\nTop feature importances:")
for name, imp in feat_imp[:10]:
    print(f"{name}: {imp:.4f}")

new_example = pd.DataFrame({
    'Area': ['Albania'],
    'Item': ['Wheat'],
    'Year': [2025],
    'average_rain_fall_mm_per_year': [1400],
    'pesticides_tonnes': [120],
    'avg_temp': [18.5]
})

pred_yield = pipeline.predict(new_example)[0]
print(f"\nPredicted Yield: {pred_yield:.2f} (units: {target_col})")

if pred_yield < 20000:
    print("Recommendation: Low yield expected. Investigate irrigation and nutrient management.")
elif 20000 <= pred_yield < 40000:
    print("Recommendation: Moderate yield. Maintain practices and monitor pests.")
else:
    print("Recommendation: Good yield expected; continue monitoring and consider premium marketing.")

```

```
joblib.dump(pipeline, "crop_yield_pipeline.joblib")
print("\nSaved trained pipeline to 'crop_yield_pipeline.joblib'")
```

Viva-ready explanations & likely questions (with short answers)

Q: Why Random Forest for yield prediction?

A: Random Forest is robust, handles non-linear interactions, tolerates unscaled features, and works well with mixed-type data. It reduces overfitting via bagging and averaging.

Q: Why OneHotEncoder instead of LabelEncoder?

A: `LabelEncoder` assigns arbitrary integers which imply ordinality. `OneHotEncoder` represents categories as independent binary features and avoids implying order. For some tree models label encoding is okay, but OHE is safer and necessary for models that assume order.

Q: What is MAE and why use it?

A: MAE (Mean Absolute Error) is the average absolute difference between predicted and true values. It's interpretable in original units and less sensitive to outliers than MSE/RMSE.

Q: What does R² tell you?

A: R² measures proportion of variance explained by the model (1 is perfect, 0 means model explains none). Use along with MAE for fuller picture.

Q: How to handle unseen categories in deployment?

A: Use encoders with `handle_unknown='ignore'` (`OneHotEncoder`) or train pipeline with `OrdinalEncoder(... handle_unknown=...)` or use embedding techniques. Always serialize pipeline to ensure same preprocessing in production.

Q: How would you tune model hyperparameters?

A: Use `GridSearchCV` or `RandomizedSearchCV` (or Bayesian opt) over `n_estimators`, `max_depth`, `min_samples_leaf`, `max_features`, etc. Use cross-validation to avoid overfitting.

Q: How to explain model predictions to farmers?

A: Use feature importances, partial dependence plots, or SHAP values to show which factors (rainfall, pesticides, temperature) drove the prediction.

Q: How do you ensure no data leakage?

A: Fit preprocessing only on training set (use Pipelines), don't use future information in features, and ensure temporal splits if data is time-series (e.g., train on earlier years, test on later years).

Q: What about time-series aspect?

A: If yield depends on temporal trends, consider time-aware splitting (train on past years, test on future years) or models like ARIMA/LSTM if sequential dependencies matter.

Q: How to improve model performance?

A: More/better features (soil nutrients, sowing date, seed variety), feature engineering (interactions), hyperparameter tuning, more data, ensemble methods, spatial models (if location coordinates available).