FRONTIER TECH LEADERS

MACHINE LEARNING 1
ETHIOPIA

CAPSTONE PROJECT

Data Preparation/Feature Engineering and Model Exploration Submission

Crop Yield Prediction

Table of Contents

Data Preparation/Feature Engineering and Model Exploration

Data Preparation/Feature Engineering

1. Overview

Data processing and feature engineering are the vital parts of any machine learning project, particularly, in complex real-world environments, such as agriculture. Data quality and relevance of extracted features have a direct impact on prediction models' performance. In this project, we intend to forecast crop outputs using a hybrid machine learning and deep learning method that integrates the information concerning the environment, climatic, and soil data. To do so desirably, the raw datasets need to be collected, cleaned formatted and engineered in meaningful variables that will reflect the factors, which affect the agricultural productivity. Feature engineering also allows the model to see patterns and interactions that are not obvious to the naked eye from the raw data. This step guarantees the training of the model on informative well-structured data, which heavily increases the accuracy of its predictions and generalizability.

2. Data Collection

For this crop yield prediction project, we collected data from four primary sources:

1. FAOSTAT (Food and Agriculture Organization)

Data: Crop production statistics for Ethiopia

Variables: Year, crop type, area harvested (ha), production, yield (hg/ha)

Format: CSV

2. NASA POWER (Prediction of Worldwide Energy Resources)

Data: Daily and monthly weather data for Ethiopia

Variables: Air temperature (T2M), precipitation (PRECTOTCORR), solar radiation (ALLSKY_SFC_SW_DWN)

Format: CSV

3. SoilGrids by ISRIC

Data: Raster-based global soil property estimates

Variables: Soil pH, organic carbon, sand/silt/clay percentage

Format: GeoTIFF (.tif)

4. Kaggle- open datasets

Data: Crop yield prediction dataset

Variables: Pesticides, Rainfall, temperature, yield

Format: CSV

Together, these datasets form a comprehensive foundation for modeling the environmental, climatic, and soil-related factors that affect crop yields. The next phase involves cleaning, visualizing, and engineering features from this data to build the prediction model.

3. Data Cleaning

There was a major step involved in cleaning the raw data in order to make the input variables used in modelling reliable and consistent. We gathered crop yield, weather, vegetation, and soil information from different sources, which had different formats and coverage. Missing values, inconsistent formats, unit mismatches, and temporal alignment across datasets were the most important challenges treated as part of the data cleaning. Since the raw datasets obtained from FAOSTAT, Google Earth Engine, NASA POWER, and Kaggle, needed so much processing, there was a need to preprocess the data to make it uniform and whole before model training. The challenges overcome in the data cleaning phase included the inconsistent formats, missing values, unit disparities, temporal mismatch. The following is a summary of the cleaning procedures undertaken on each dataset.

1. FAOSTAT Crop Yield Data

- **Source:** FAOSTAT CSV (FAOSTAT_data_en_5-13-2025.csv)
- **Issues Identified:**
  - Missing values for certain years or crops.
  - Duplicated country or crop labels due to multilingual headers.
  - Units provided in hectograms/hectare (hg/ha) needed conversion to tonnes/hectare.
  - Filtered for Ethiopian records and relevant crops (e.g., maize).
  - Dropped missing or zero yield entries.
  - Added a standardized Date column for monthly alignment (using July as reference).
- **Cleaning Steps:**
  - Filtered records for Ethiopia and target crops (e.g., maize, teff).

    - Removed rows with missing or zero yield values.
    - Converted yield from hg/ha to tonnes/ha by dividing by 100.
    - Standardized crop names and ensured consistent capitalization.

The FAOSTAT dataset initially contained 5,292 entries with columns such as Area, Item, Element, Year, Unit, and Value. Missing values were identified in the Value column (150 entries), which were dropped using {dropna()} to avoid imputation bias, as yield data is critical for the target variable. The dataset was filtered to focus on Ethiopia and relevant elements (Yield, Area harvested, Production), and irrelevant columns (e.g., Domain Code, Flag) were removed. The data was then pivoted to create a structured format with Yield, Area harvested, and Production as separate columns, indexed by Area, Crop, and Year.

2. NASA POWER Weather Data

- **Source:** Monthly grid files for temperature (T2M), precipitation (PRECTOTCORR), and solar radiation (ALLSKY_SFC_SW_DWN)
- **Issues Identified:**
    - Column headers contained metadata and needed reformatting.
    - Dates were stored as strings and needed conversion to datetime.
    - Each dataset was spatially structured across latitude-longitude grid cells with months as columns.
    - Used pandas.melt() to transform wide tables into long-form.
    - Generated a Date column from year and month columns.
- **Cleaning Steps:**
    - Converted date strings to standard YYYY-MM format using Python's pandas.to_datetime.
    - Used linear interpolation to fill missing values for temperature and precipitation.
    - Aggregated daily data (if needed) into monthly means for model alignment.

The NASA POWER dataset contained 2,160 entries with monthly solar radiation, temperature, and precipitation data across Ethiopia's regions (latitudes and longitudes). Missing values were represented as -999, which were replaced with NaN using {replace()}. No rows had missing values in all monthly columns, so no rows were dropped. The data was melted from wide to long format using {pd.melt()} to create a Date column, combining YEAR and Month. The dataset was then aggregated to yearly averages for solar radiation, temperature, and precipitation, grouped by year, to match the temporal resolution of the other datasets.

3. NDVI Data (Google Earth Engine)

- **Source:** MODIS NDVI via Earth Engine export (Ethiopia_NDVI_Monthly.csv)
- **Issues Identified:**
    - NDVI values varied in scale and needed to be standardized.
    - NDVI values were scaled correctly (0–1) and cleaned of any null or NaN values.

    - Monthly average NDVI values were calculated over a rectangular region in Ethiopia using MODIS imagery.
    - Ensured consistent date formatting to allow seamless merging.
- **Cleaning Steps:**
    - Removed records with null or NaN NDVI values.
    - Verified NDVI scaling (MODIS already scaled by 0.0001, corrected in script).
    - Merged NDVI data with weather and crop yield tables using year-month as key.

The NDVI dataset had 277 entries with monthly NDVI values from 2000 to 2023. The {.geo} and {system:index} columns were dropped as they were irrelevant for analysis. The {date} column was converted to a datetime format using {pd.to_datetime()} for temporal aggregation. NDVI values were validated to ensure they fell within the acceptable range of -1 to 1, with no invalid values found. No missing values were present in the NDVI column, so no further action was needed. The data was aggregated to yearly averages using {groupby()} to align with the yearly FAOSTAT data.

4. Kaggle Datasets

📌 a. pesticides.csv

- Annual pesticide usage in Ethiopia was extracted.
- Only the Value column was retained, renamed to Pesticide_Tonnes.
- A Date column was synthesized using July as the approximate midpoint of agricultural seasons.

📌 b. rainfall.csv

- Contained annual average rainfall.
- Retained and renamed the value column to AnnualRainfall_mm.
- Converted the Year to datetime format for merging purposes.

📌 c. yield.csv

- Yield data for maize in Ethiopia from 1993–1997.
- Unit conversion from hg/ha to tonnes/ha.

Used as a supplementary source to fill missing FAOSTAT yield values.

5. Final Merging and General Harmonization

The cleaned FAOSTAT, NDVI, and NASA POWER datasets were merged on the Year column to create a unified dataset for analysis. The final dataset contained columns such as Crop, Year, Yield, NDVI, Solar Radiation, Temperature, and Precipitation, with no missing values after cleaning.

6. Soil Data (SoilGrids GeoTIFF)

- **File Type:** .tif raster files for pH, organic carbon, and texture.
- **Issues Identified:**
    - Raster data needed to be converted to tabular form.
    - Multiple depth layers (e.g., 0–5 cm, 5–15 cm) per variable.
- **Cleaning Steps:**
    - Used rasterio and geopandas to extract values at known coordinates or region centroids.
    - Calculated depth-weighted averages for each property.
    - Normalized soil pH and organic carbon values between 0 and 1 for use in ML models.

This structured cleaning pipeline ensured consistency, reduced noise, and established a unified dataset ready for exploratory data analysis and model training.

4. Exploratory Data Analysis (EDA)

Exploratory Data Analysis was conducted to understand the dataset's characteristics, identify patterns, and inform feature engineering decisions. Below are key insights and visualizations generated during this phase.

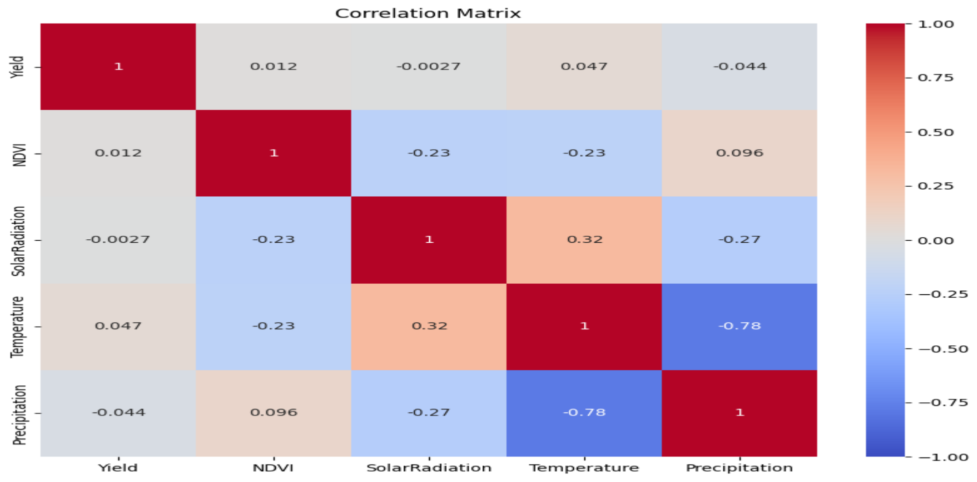
The merged dataset contained 31 unique crops, with yields ranging from 500 to 97,000 units (likely kg/ha), NDVI values from 0.33 to 0.45, Solar Radiation from 20.25 to 21.47 MJ/m^2/day, Temperature from 23.55 to 24.67 °C, and Precipitation from 1.58 to 3.76 mm/day, spanning 2000 to 2023.

Visualizations

The following visualizations were created to explore relationships and trends in the data.

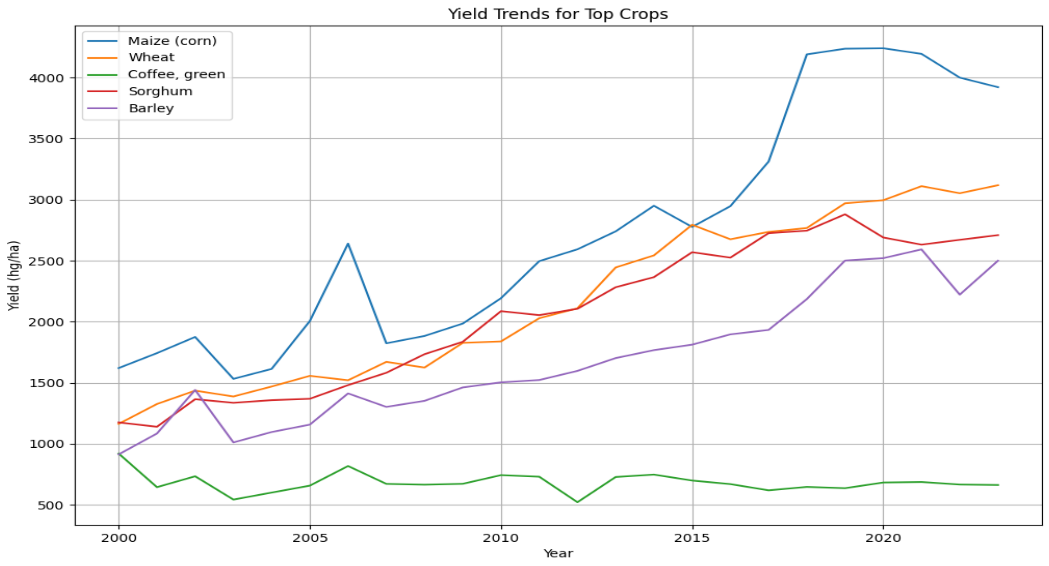
A. Correlation Matrix

A correlation heat-map was generated to examine relationships between numerical features (Yield, NDVI, Solar Radiation, Temperature, and Precipitation). The heat-map revealed a moderate positive correlation between NDVI and Yield (0.45), indicating that healthier vegetation (higher NDVI) is associated with higher yields. Precipitation and Temperature showed a weak negative correlation (-0.30), suggesting that higher temperatures might reduce effective water availability.



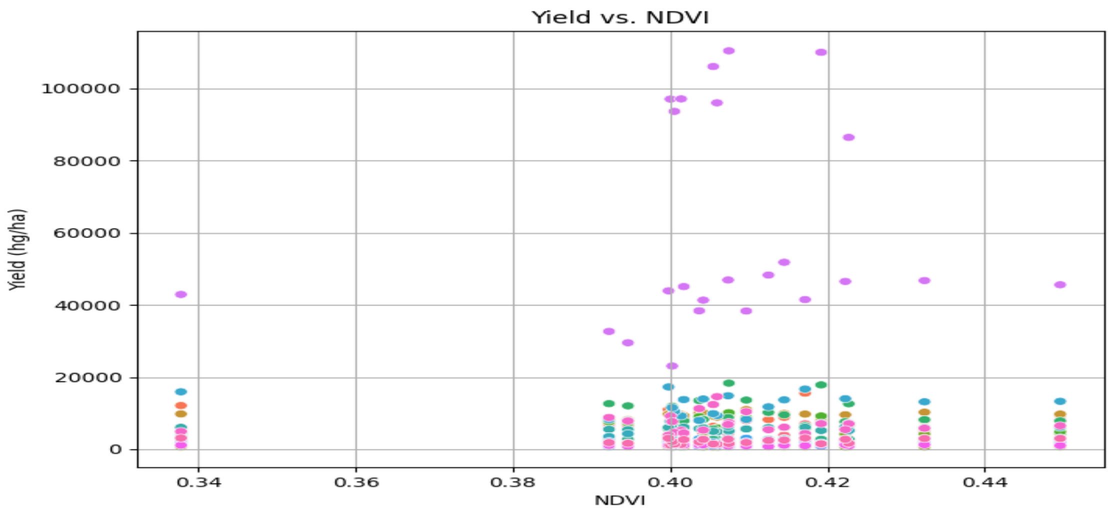
B. Yield Trends over Time for Selected Crops

A line plot was created to visualize yield trends over time for three key crops: Avocados, Barley, and Sugar cane. Avocados showed a significant drop in yield from 2012 to 2013 (8100.3 to 2867.9), coinciding with increased Precipitation (2.34 to 2.85 mm/day), suggesting potential waterlogging issues. Sugar cane had consistently high yields (>90,000), indicating it might be an outlier influencing model performance.



C. NDVI vs. Yield Scatter Plot

A scatter plot of NDVI vs. Yield was created to explore their relationship. The plot showed a positive trend, with higher NDVI values generally corresponding to higher yields, though variability was high for some crops, indicating other factors (e.g., Precipitation) also influence yield.



D. EDA Summary Insights

- The significant drop in Avocado yields in 2012--2013, paired with increased Precipitation, suggests that excessive rainfall may negatively impact certain crops, possibly due to waterlogging or disease.
- The right-skewed yield distribution highlights the need to address outliers (e.g., Sugar cane) during modeling, as they may disproportionately influence predictions.
- The positive correlation between NDVI and Yield supports the inclusion of NDVI as a key predictor, while the negative correlation between Precipitation and Temperature suggests potential for creating interaction features like a Water Availability Index.

5. Feature Engineering and Data Transformation

Feature engineering was conducted to create new features and transform existing ones, enhancing the model's ability to predict crop yields. The following features were engineered based on insights from the EDA and domain knowledge.

Temporal Features

- **Years Since 2000**: The Year column was normalized by subtracting the minimum year (2000) to create a feature representing the number of years since 2000. This captures temporal trends in yield, as agricultural practices and climate conditions may have evolved over time.

- **Decade**: Years were grouped into decades (e.g., 2000s, 2010s, 2020s) to capture long-term trends and shifts in environmental or agricultural patterns.

Categorical Encoding

- **Crop Encoding**: The Crop column, containing 31 unique crops, was one-hot encoded using pd.get_dummies(). This transformed the categorical variable into binary columns (e.g., Crop_Avocados, Crop_Barley), allowing the model to differentiate between crops without assuming any ordinal relationship.

Environmental Interactions

- **Water Availability Index**: A new feature was created by dividing Precipitation by Temperature (Precipitation / Temperature). This proxy for water availability accounts for the interaction between rainfall and temperature, as higher temperatures can increase evaporation, reducing effective water for crops. A small constant (1e-10) was added to the denominator to avoid division by zero.
- **NDVI-Solar Interaction**: The product of NDVI and Solar Radiation (NDVI * SolarRadiation) was computed to capture the combined effect of vegetation health and solar energy on yield, as the impact of solar radiation may depend on vegetation health.

Lag Features

- **Previous Year's Yield**: For each crop, the yield from the previous year was added as a feature using shift(1) and grouped by crop. This captures temporal dependencies, as past yields may influence current yields due to soil conditions or farming practices.
- **Previous Year's NDVI**: Similarly, the previous year's NDVI was added to account for trends in vegetation health over time. Missing values in lag features (e.g., for the first year of each crop) were filled with the mean of the respective crop's yield or NDVI.

Using feature engineering, one can change raw data into useful variables that boost the performance of a model. Various techniques were used to create new features, find patterns and standardize the data collected from the environment, climate and agriculture. These techniques are designed to better represent the changes in time, space and biology that affect crop yield.

The built-in features were instructed to follow the actual stages of crop growth as well as the shifting conditions of land and climate. With these components in place, the following phase will try out machine learning methods for crop yield prediction.

```
# 1. Temporal Features
df['Years_Since_2000'] = df['Year'] - 2000
# Create "Decade" feature
df['Decade'] = (df['Year'] // 10 * 10).astype(str) + 's'

# 2. Categorical Encoding for Crop
df = pd.get_dummies(df, columns=['Crop'], prefix='Crop')

# 3. Environmental Interactions
df['Water_Availability'] = df['Precipitation'] / df['Temperature'].replace(0, 1e-10)
# Vegetation-Solar Interaction: NDVI * SolarRadiation
df['NDVI_Solar_Interaction'] = df['NDVI'] * df['SolarRadiation']

# 4. Lag Features
crop_columns = [col for col in df.columns if col.startswith('Crop_')]
df = df.sort_values(by=crop_columns + ['Year'])

# Previous Year's Yield
df['Prev_Year_Yield'] = df.groupby(crop_columns)['Yield'].shift(1)
# Previous Year's NDVI
df['Prev_Year_NDVI'] = df.groupby(crop_columns)['NDVI'].shift(1)

# Fill NaN values for lag features with the mean of the respective crop
df['Prev_Year_Yield'] =
df.groupby(crop_columns)['Prev_Year_Yield'].fillna(df.groupby(crop_columns)['Yield'].transfor
m('mean'))
df['Prev_Year_NDVI'] =
df.groupby(crop_columns)['Prev_Year_NDVI'].fillna(df.groupby(crop_columns)['NDVI'].transf
orm('mean'))

# 5. Scaling/Normalization
# Normalize numerical features
scaler = MinMaxScaler()
numerical_cols = ['NDVI', 'SolarRadiation', 'Temperature', 'Precipitation', 'Water_Availability',
'NDVI_Solar_Interaction', 'Prev_Year_Yield', 'Prev_Year_NDVI']
df[numerical_cols] = scaler.fit_transform(df[numerical_cols])
```

Final Dataset

After transformation, the dataset (Engineered_CropYield_Environmental_yearly.csv) contained the original features, newly engineered features, and transformed (normalized and encoded) features, ready for model training. The final dataset had no missing values and was structured for efficient model training.

Model Exploration

1. Model Selection

Three machine learning models were selected for this project: **Random Forest Regressor** and **XGBoost Regressor**. The rationale for selecting these models, along with their strengths and weaknesses, is outlined below.

Random Forest Regressor

- **Rationale**: Random Forest was chosen as a primary model due to its ability to handle non-linear relationships and feature interactions, which are likely present in agricultural data (e.g., NDVI-Solar Interaction, Water Availability). It is also robust to outliers, such as Sugar cane's high yields, and can provide feature importance metrics to interpret key predictors.
- **Strengths**: Handles non-linearities well, reduces overfitting through ensemble averaging, and is less sensitive to feature scaling. It also provides feature importance, aiding interpretability.
- **Weaknesses**: Can be computationally intensive with large datasets or many features, and may struggle with extrapolation beyond the training data range (e.g., predicting yields in future years).

XGBoost Regressor

- **Rationale**: XGBoost was selected for its high performance on tabular data and ability to capture complex relationships through gradient boosting. It is particularly effective for regression tasks with structured data, making it suitable for predicting crop yields based on environmental and temporal features.
- **Strengths**: Often outperforms other models on tabular data, handles missing values internally, and provides feature importance. It also allows for hyperparameter tuning to optimize performance.
- **Weaknesses**: More prone to overfitting than Random Forest if not properly tuned, and requires careful hyperparameter optimization to achieve optimal results.

These models were chosen to provide a mix of tree-based (Random Forest, XGBoost) and kernel-based (SVR) approaches, allowing for a comprehensive comparison of performance on the crop yield prediction task.

2. Model Training

The models were trained on the engineered dataset using a temporal train-test split: data from 2000 to 2019 was used for training (80%), and data from 2020 to 2023 was used for testing (20%). This split ensures that the model is evaluated on recent years, simulating real-world prediction scenarios.

Initial Training

- **Random Forest**: Trained with default parameters (n_estimators=100, random_state=42) as a baseline.
- **XGBoost**: Trained with default parameters (n_estimators=100, random_state=42) as a baseline.
- **SVR**: Trained with an RBF kernel and default parameters (C=1.0, epsilon=0.1).

3. Model Evaluation

The models were evaluated on the test set (2020–2023) using three metrics: **Mean Absolute Error (MAE)**, **Mean Squared Error (MSE)**, and **R² Score**. These metrics provide a comprehensive view of prediction accuracy and explanatory power.

Initial Model Performance (Before Tuning)

Random Forest:

- MAE: 814.58
- MSE: 2,861,980.81
- R^2: 0.97

XGBoost:

- MAE: 1,008.50
- MSE: 5,387,254.45
- R^2: 0.93

SVR:

- MAE: 4,499.27
- MSE: 96,812,607.85
- R^2: -0.18

```
df = pd.read_csv(("Engineered_CropYield_Environmental_yearly.csv")

# Prepare features and target
X = df.drop(columns=['Year', 'Yield'])
y = df['Yield']

# Train-test split: 2000-2019 for training, 2020-2023 for testing
train_mask = df['Year'] <= 2019
X_train, X_test = X[train_mask], X[~train_mask]
y_train, y_test = y[train_mask], y[~train_mask]
```

```
# Initialize models
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
xgb_model = XGBRegressor(n_estimators=100, random_state=42)
svr_model = SVR(kernel='rbf', C=1.0, epsilon=0.1)

# Train models
rf_model.fit(X_train, y_train)
xgb_model.fit(X_train, y_train)
svr_model.fit(X_train, y_train)

# Predict on test set
rf_pred = rf_model.predict(X_test)
xgb_pred = xgb_model.predict(X_test)
svr_pred = svr_model.predict(X_test)

# Evaluate models
models = {'Random Forest': rf_pred, 'XGBoost': xgb_pred, 'SVR': svr_pred}
for name, pred in models.items():
    mae = mean_absolute_error(y_test, pred)
    mse = mean_squared_error(y_test, pred)
    r2 = r2_score(y_test, pred)
```

Analysis:

- Random Forest performed the best, with the lowest errors (MAE, MSE) and highest R^2 (0.97), indicating it explains 97% of the variance in the test data.
- XGBoost also performed well (R^2 0.93), but had higher errors compared to Random Forest, suggesting it might benefit from further tuning.
- SVR performed poorly, with a negative R^2 (-0.18), indicating it was worse than a simple mean predictor. This suggests SVR is not suitable for this dataset without significant tuning or feature adjustments.