Smart Crop Prediction

A Comprehensive Analysis of Machine Learning Models"

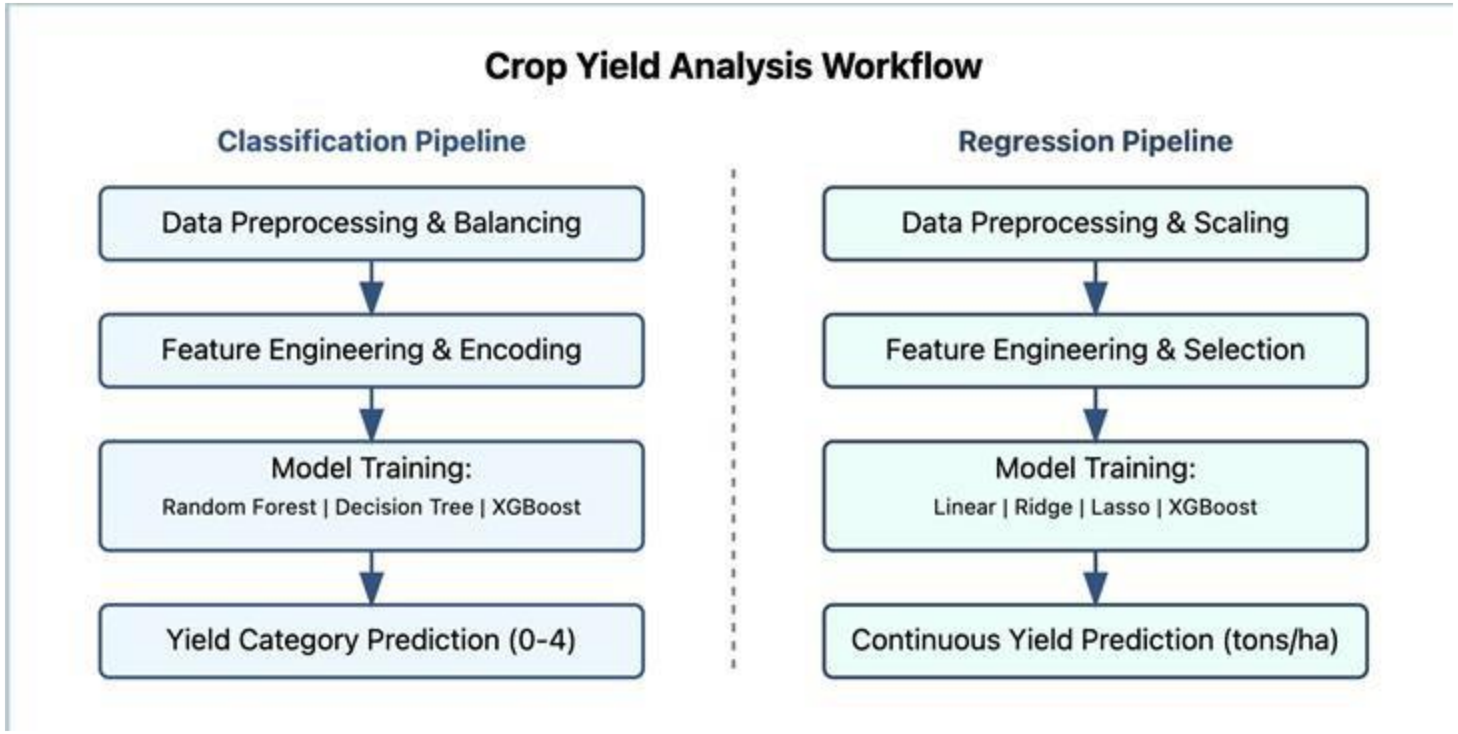
Ajay K Mashapari

Outline

- Project Duration: 2024
- Models Analyzed: 15+ across classification and regression
- Dataset Features: Weather, Soil, Agricultural Practices

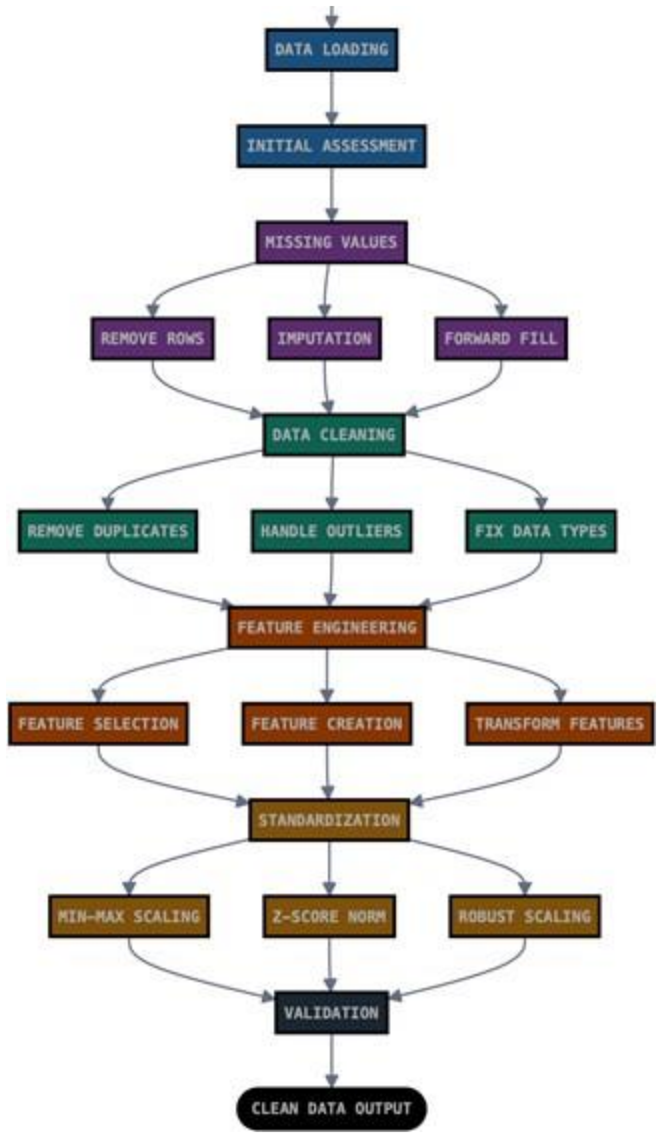
Project Overview

- **Objectives & Scope:**
  - Classification Goals:
    - Predict yield categories (0-4)
    - Identify key performance drivers
    - Achieve >80% accuracy
  - Regression Goals:
    - Precise yield prediction in tons/hectare
    - RMSE target < 0.5
    - R^2 target > 0.9



Data Preprocessing Steps

- **Data Cleaning & Preparation:**
  - Feature Engineering:
    - Standardization of numeric features
    - One-hot encoding for categorical variables
    - SMOTE for class balancing
  - Features Processed:
    - Numeric: Rainfall, Temperature, Days to Harvest
    - Categorical: Region, Soil Type, Crop, Weather
    - Binary: Fertilizer, Irrigation



Imbalance issue

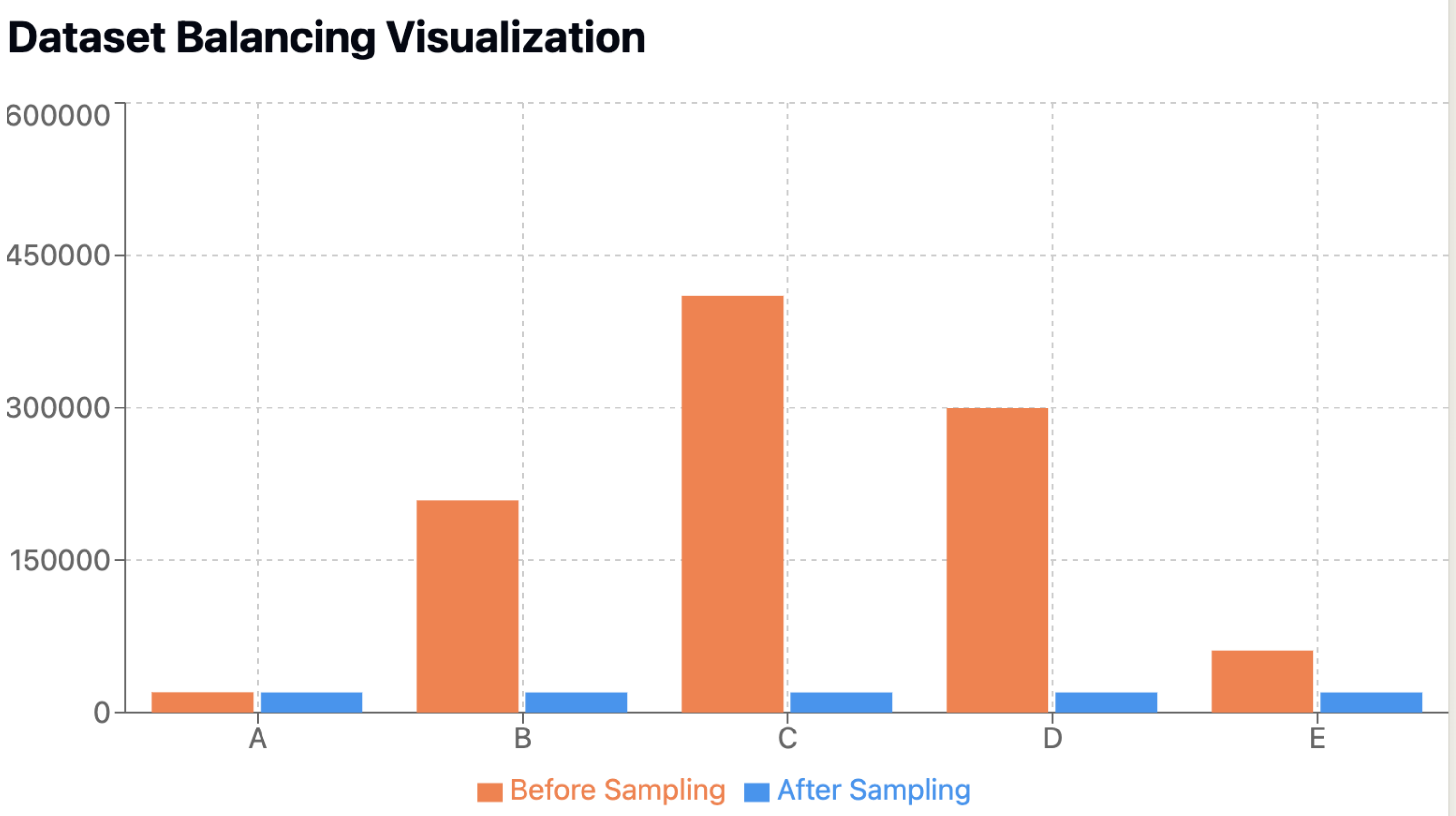
```
# Count occurrences
value_counts = df_C['yield_category'].value_counts()
counts = value_counts[['A','B', 'C', 'D', 'E']]

# Print the counts
print("Value Counts:")
print(counts)
```

```
Value Counts:
yield_category
A     20183
B    208676
C    409999
D    299923
E     60988
Name: count, dtype: int64
```

Solution:Undersampling

```python
import pandas as pd
import numpy as np

# Set random seed for reproducibility
np.random.seed(42)

# Sample 20000 from each category
df_C_bal = pd.concat([
    df_C[df_C['yield_category'] == 'A'].sample(n=20000, random_state=42),
    df_C[df_C['yield_category'] == 'B'].sample(n=20000, random_state=42),
    df_C[df_C['yield_category'] == 'C'].sample(n=20000, random_state=42),
    df_C[df_C['yield_category'] == 'D'].sample(n=20000, random_state=42),
    df_C[df_C['yield_category'] == 'E'].sample(n=20000, random_state=42)
])

# Verify the new counts
print("New Value Counts:")
print(df_C_bal['yield_category'].value_counts())

# Visualize the balanced distribution
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
df_C_bal['yield_category'].value_counts().plot(kind='bar')
plt.title('Balanced Distribution of Categories')
plt.xlabel('Category')
plt.ylabel('Count')
plt.tight_layout()
plt.show()

# Save the balanced dataset if needed
# balanced_df_num.to_csv('balanced_dataset.csv', index=False)
```

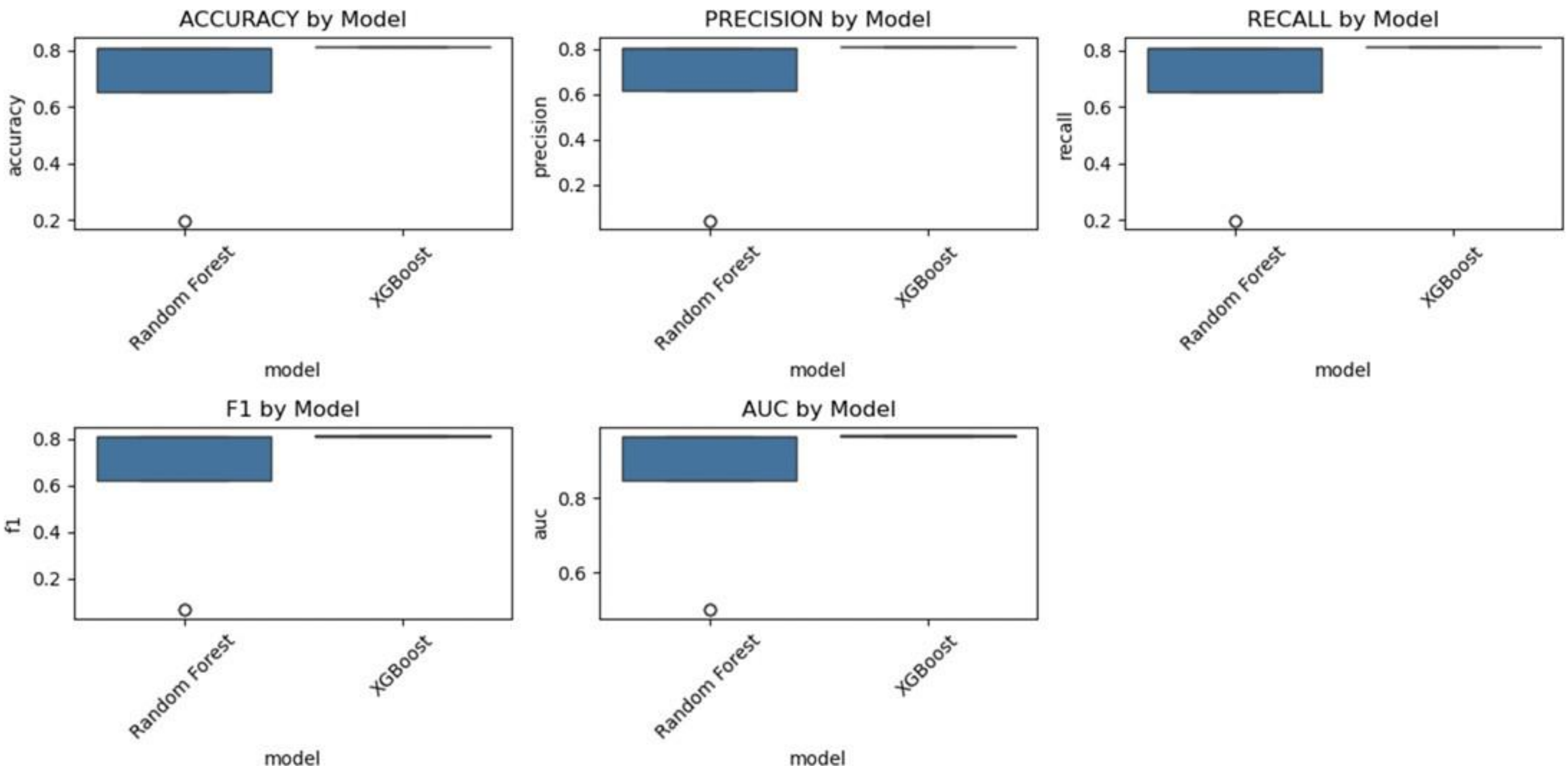
```
New Value Counts:
yield_category
A    20000
B    20000
C    20000
D    20000
E    20000
Name: count, dtype: int64
```



Balancing method:
Original dataset had significant imbalance (Category C: 409,999 vs Category A: 20,183)
Used random sampling to extract exactly 20,000 samples from each category
Resulted in perfectly balanced dataset with equal representation

Model Selection(Classification)

- Models Evaluated:
  - Decision Trees
  - Random Forest
  - XGBoost
- Comparison Criteria:
  - Accuracy, Precision, Recall
  - F1-Score, AUC-ROC

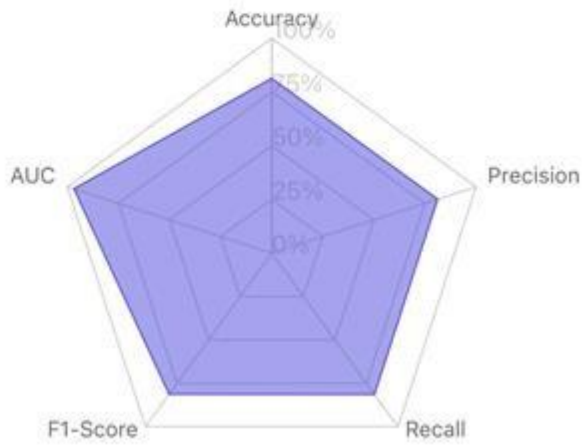

ACCURACY by Model
accuracy
0.2
0.4
0.6
0.8
Random Forest
XGBoost
model
PRECISION by Model
precision
0.2
0.4
0.6
0.8
Random Forest
XGBoost
model
RECALL by Model
recall
0.2
0.4
0.6
0.8
Random Forest
XGBoost
model
F1 by Model
f1
0.2
0.4
0.6
0.8
Random Forest
XGBoost
model
AUC by Model
auc
0.6
0.8
Random Forest
XGBoost
model

Classification Analysis - Model Performance

- XGBoost Configuration (Best Model):
- Parameters:
  - n_estimators=100
  - subsample=0.9 Performance Metrics:
- Accuracy: 81.35%
- Precision: 81.19%
- Recall: 81.35%
- F1-Score: 81.24%
- AUC: 0.9681

XGBoost Model Performance Metrics

Configuration: n_estimators=100, subsample=0.9



Accuracy **81.35%**	Precision **81.19%**
Recall **81.35%**	F1-Score **81.24%**
AUC **96.81%**	

Average Metrics by Model Type

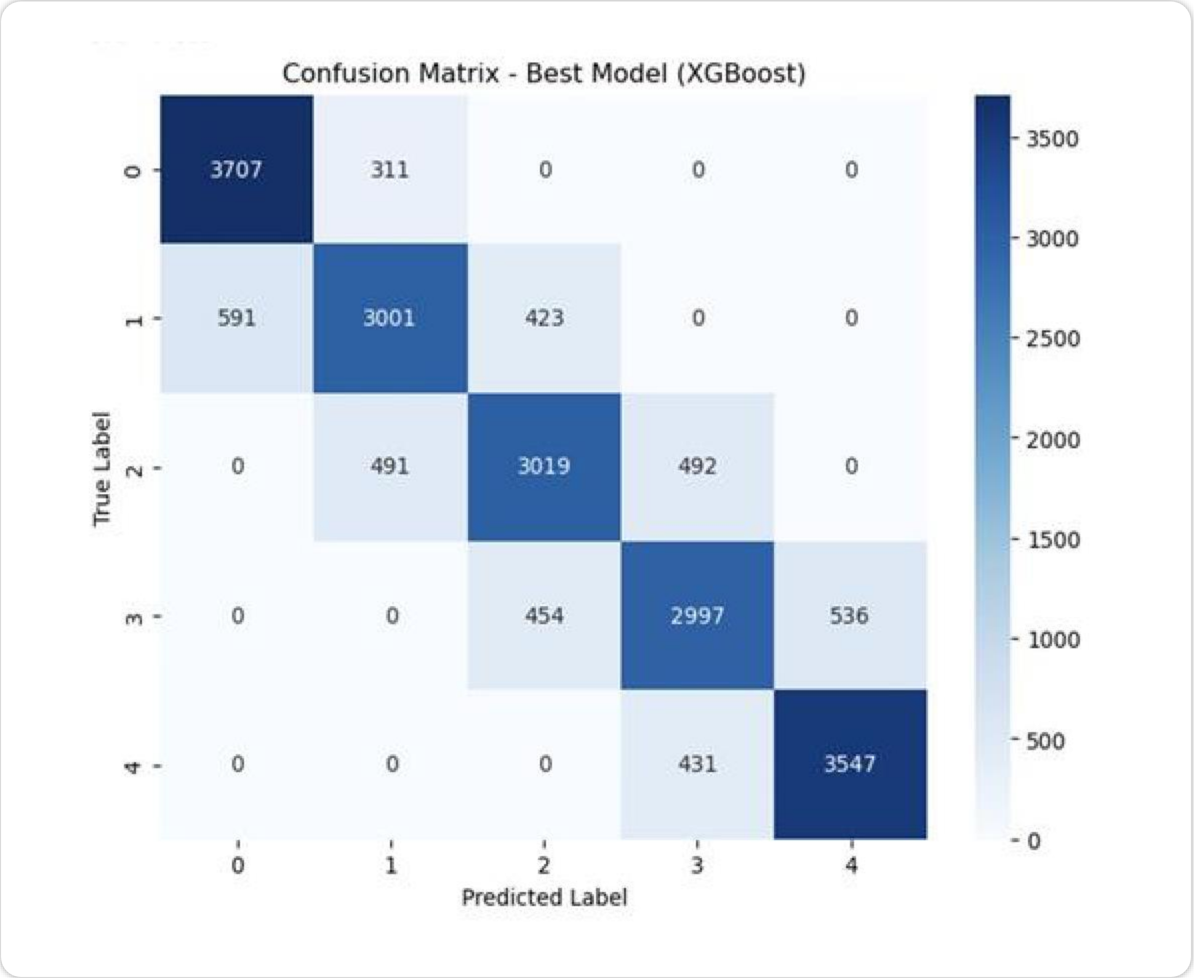
Model	Accuracy	Precision	Recall	F1	AUC
Random Forest	0.6554	0.6150	0.6554	0.6213	0.8482
XGBoost	0.8125	0.8110	0.8125	0.8113	0.9679

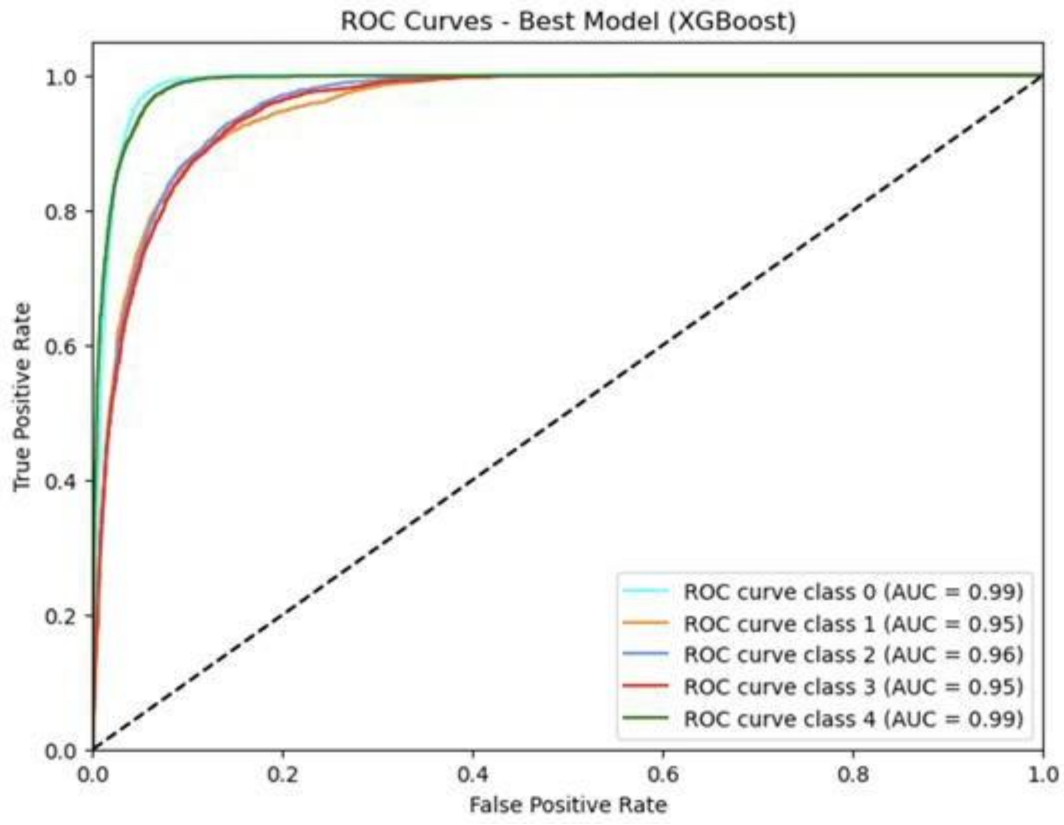
Best Model Performance by Metric

Metric	Model	Parameters	Score	ID
Accuracy	XGBoost	n_estimators: 100, subsample: 0.9	0.8136	19
Precision	XGBoost	n_estimators: 100, subsample: 0.9	0.8119	19
Recall	XGBoost	n_estimators: 100, subsample: 0.9	0.8136	19
F1	XGBoost	n_estimators: 100, subsample: 0.9	0.8124	19
AUC	XGBoost	n_estimators: 100, subsample: 0.7	0.9682	17

Classification Results - Detailed Analysis

1. Confusion Matrix:
   1. Strong diagonal pattern
   2. Minimal cross-category confusion
2. ROC Analysis:
   1. Class 0: AUC = 0.99
   2. Class 1: AUC = 0.95
   3. Class 2: AUC = 0.96
   4. Class 3: AUC = 0.95
   5. Class 4: AUC = 0.99

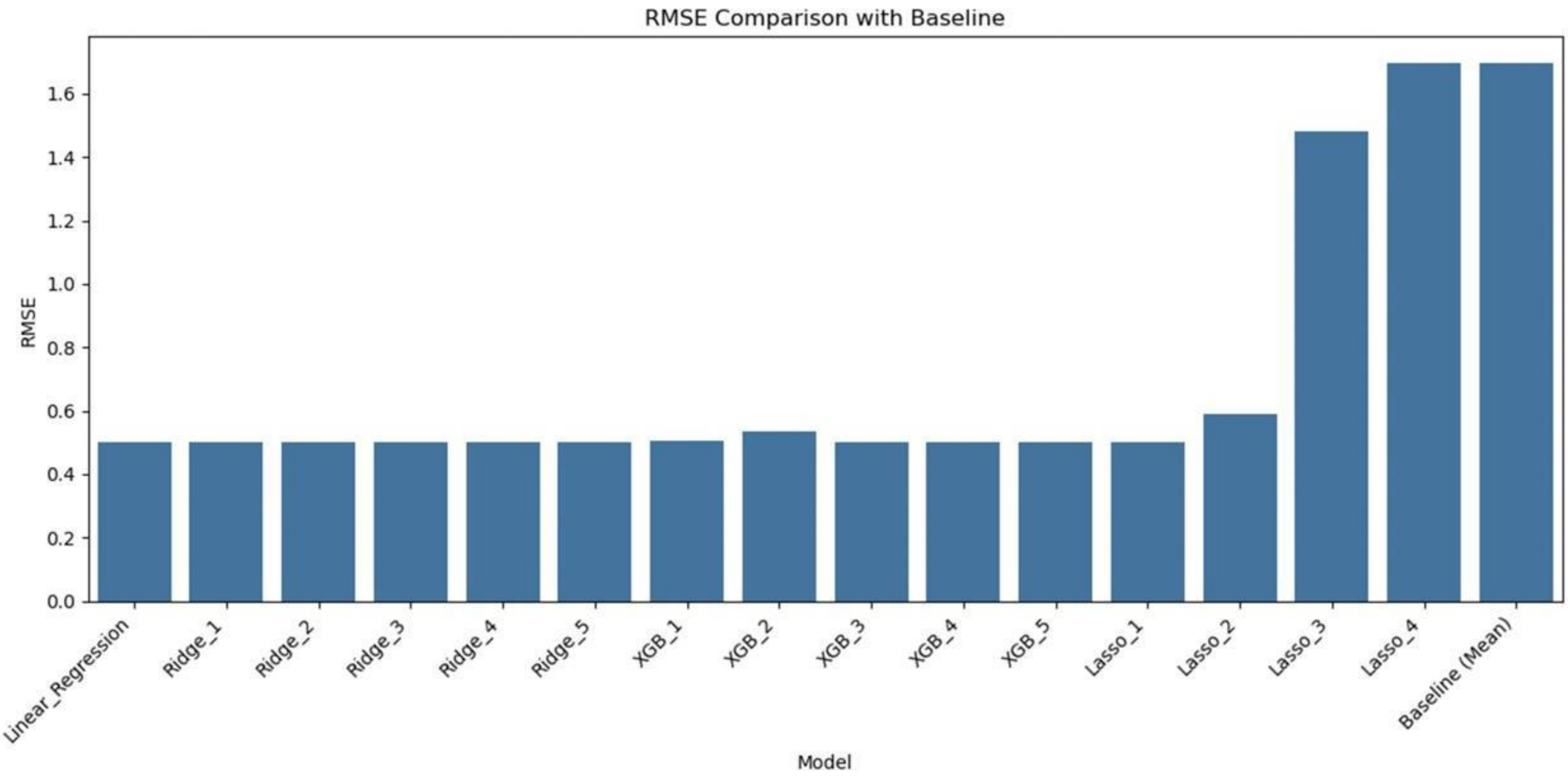

Confusion Matrix - Best Model (XGBoost)
True Label
Predicted Label
0
1
2
3
4
3707
311
591
3001
423
491
3019
492
454
2997
536
431
3547
3500
3000
2500
2000
1500
1000
500


ROC Curves - Best Model (XGBoost)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve class 0 (AUC = 0.99)
ROC curve class 1 (AUC = 0.95)
ROC curve class 2 (AUC = 0.96)
ROC curve class 3 (AUC = 0.95)
ROC curve class 4 (AUC = 0.99)

Regression Model Comparison

- Model Performance Rankings:

1. Ridge Regression (Best):
   1. RMSE: 0.499271
   2. R^2: 0.913234
2. Linear Regression:
   1. Baseline performance
3. XGBoost Variations:
   1. 5 different configurations
4. Lasso Regression:
   1. 4 different alpha values


RMSE Comparison with Baseline
RMSE
Model
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
Linear_Regression
Ridge_1
Ridge_2
Ridge_3
Ridge_4
Ridge_5
XGB_1
XGB_2
XGB_3
XGB_4
XGB_5
Lasso_1
Lasso_2
Lasso_3
Lasso_4
Baseline (Mean)

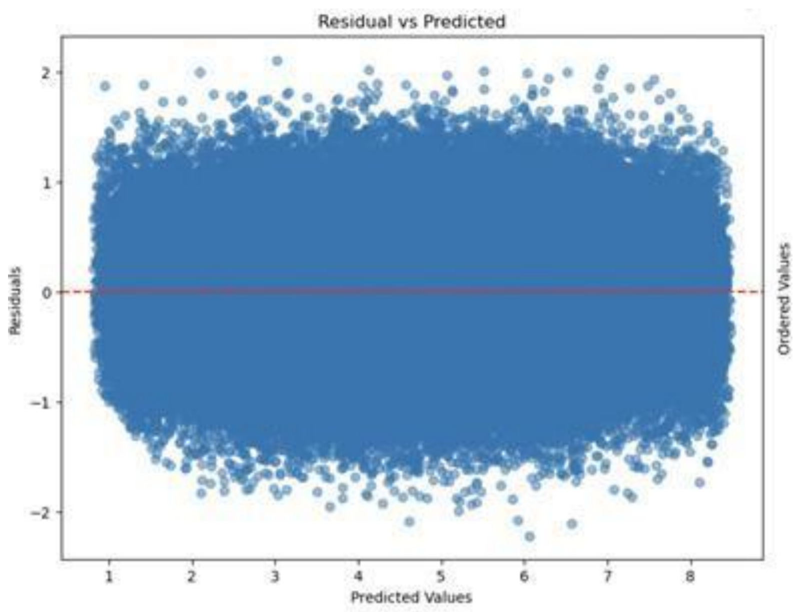
Best Regression Model Analysis

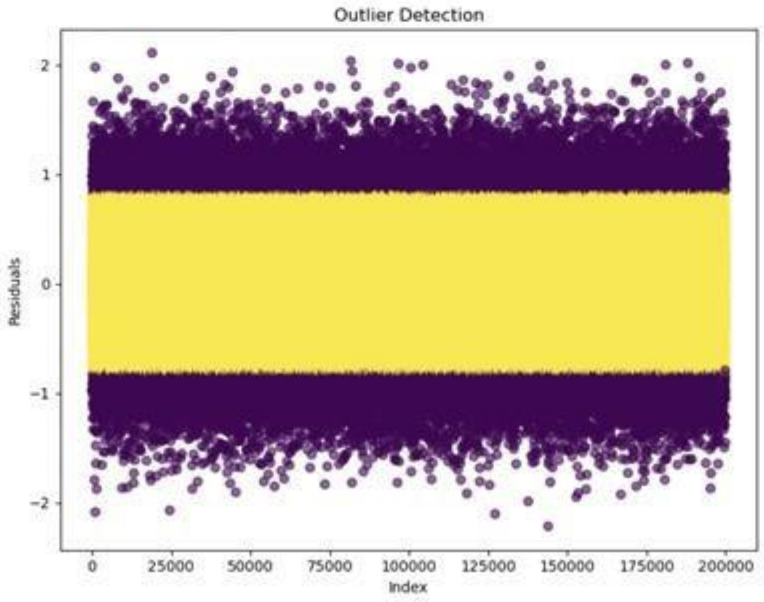
- Ridge Regression Details:
- Configuration: alpha=100.0
- Performance Metrics:
  - RMSE: 0.499271
  - R^2: 0.913234
  - CV Mean R^2: 0.912958
  - CV Std R^2: 9.43e-05

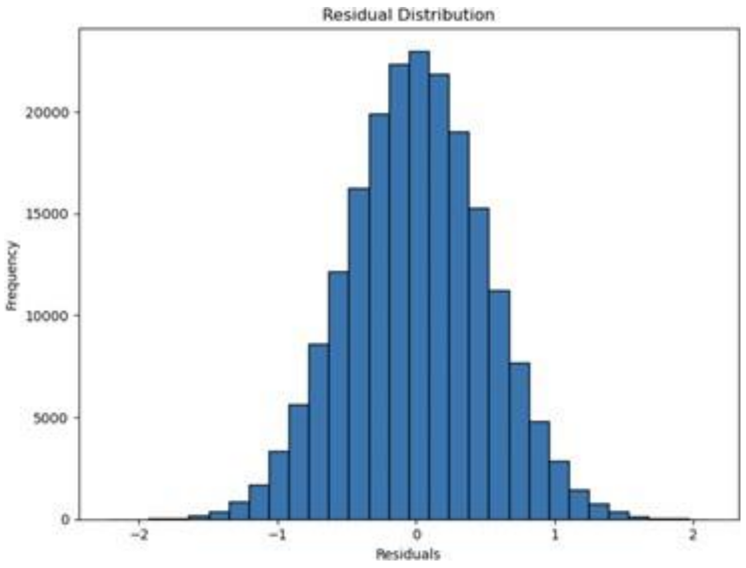
Regression Diagnostics

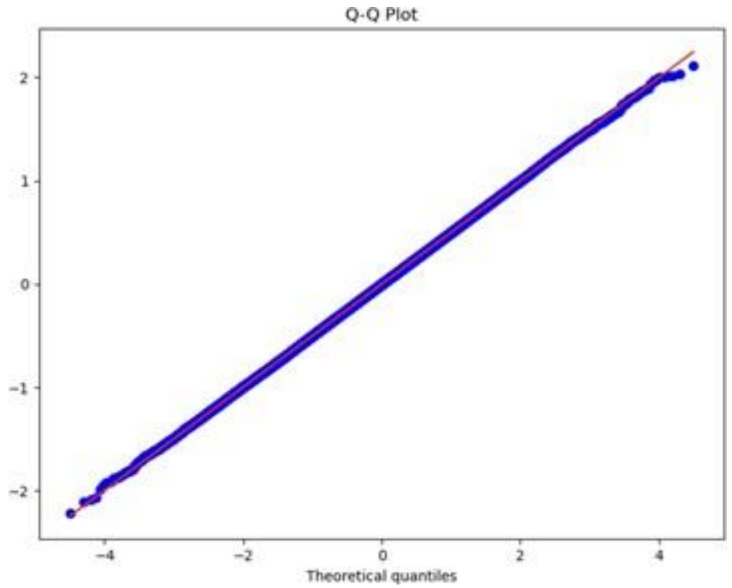
Detailed Analysis:

1. Residual Plot:
   1. Random scatter around zero
   2. Consistent spread
2. Outlier Detection:
   1. < 10% outliers identified
3. Residual Distribution:
   1. Normal distribution
   2. Centered at zero
4. Q-Q Plot:
   1. Strong diagonal alignment


Residual vs Predicted
Residuals
Ordered Values
Predicted Values
2
1
0
−1
−2
1
2
3
4
5
6
7
8


Outlier Detection
Residuals
Index
2
1
0
−1
−2
0
25000
50000
75000
100000
125000
150000
175000
200000


Residual Distribution
Frequency
Residuals
20000
15000
10000
5000
0
−2
−1
0
1
2


Q-Q Plot
Theoretical quantiles
2
1
0
−1
−2
−4
−2
0
2
4

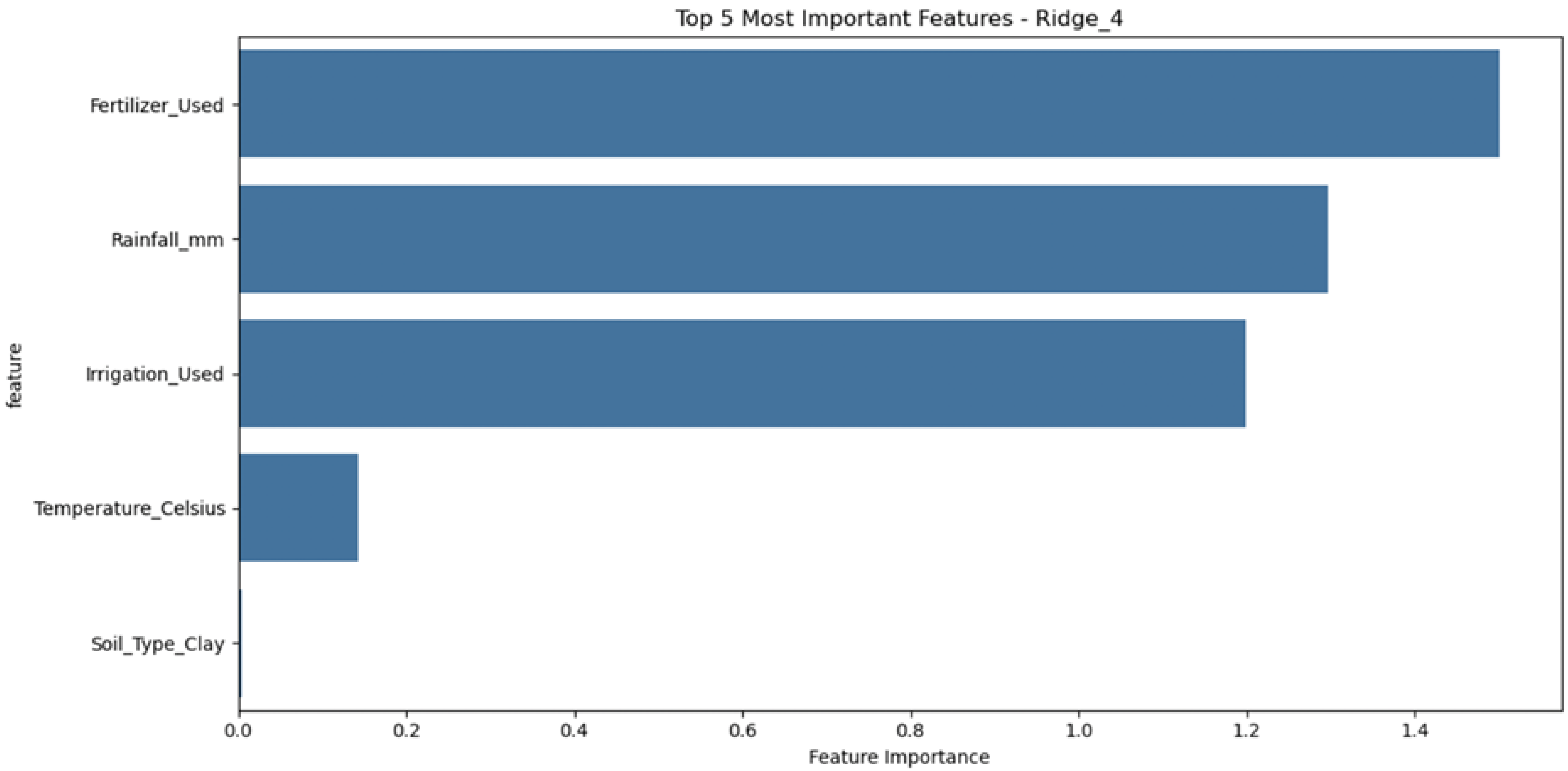
Feature Importance Analysis

Key Drivers of Yield:

1. Fertilizer Usage (1.4 score)
2. Rainfall (1.3 score)
3. Irrigation (1.2 score)
4. Temperature (0.2 score) Impact Analysis:

- Agricultural inputs dominate
- Weather factors significant
- Soil characteristics moderate impact

Feature Importance Analysis

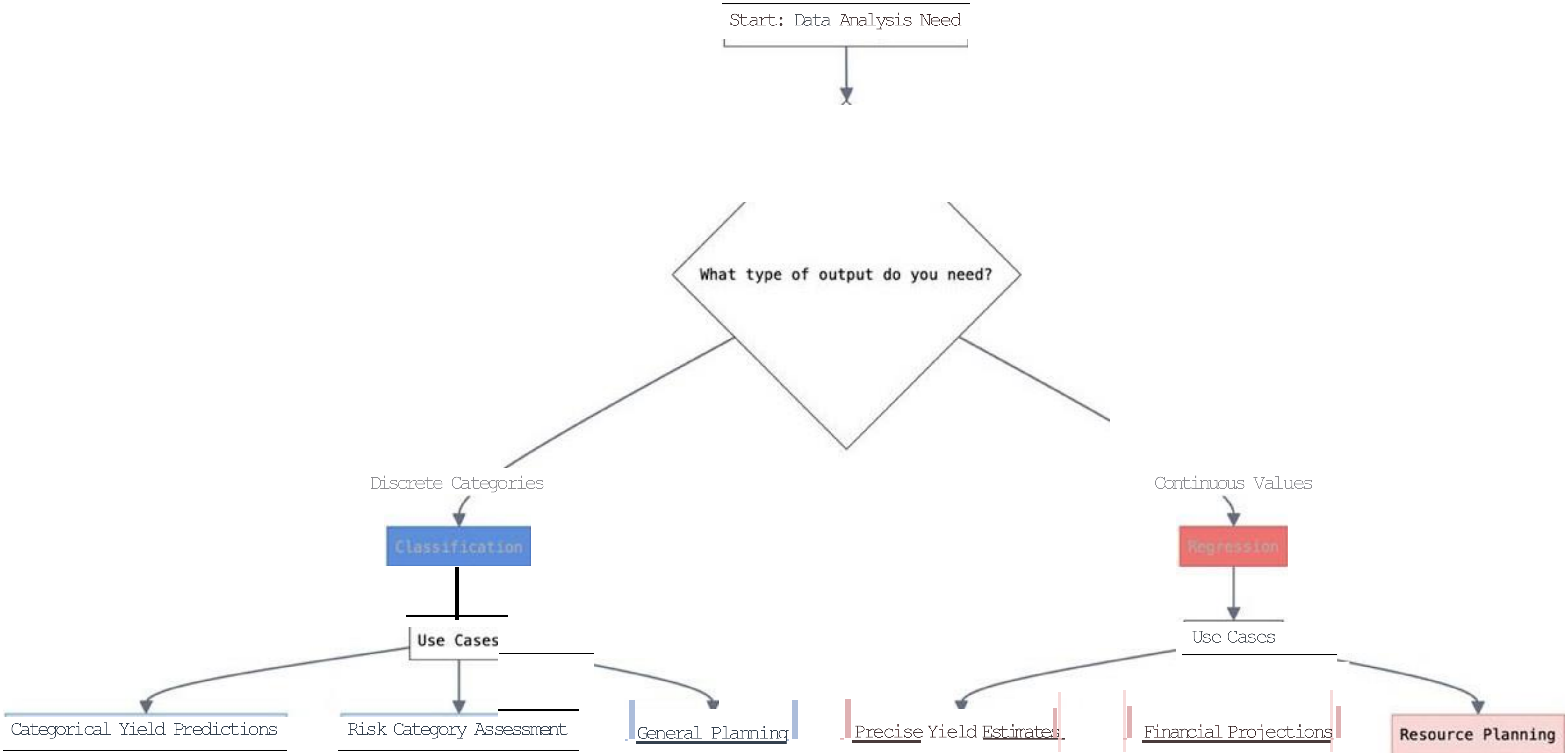


Real-World Applications

1. Agricultural Planning:
    1. Yield prediction accuracy: ±0.5 tons/hectare
    2. 91.32% variance explained
2. Financial Planning:
    1. Revenue forecasting
    2. Risk assessment
3. Resource Optimization:
    1. Input planning
    2. Resource allocation
4. Market Intelligence:
    1. Supply prediction
    2. Price forecasting

Model Selection Guidelines

- When to Use Classification:
- Categorical yield predictions
- Risk category assessment
- General planning
- When to Use Regression:
- Precise yield estimates
- Financial projections
- Resource planning


Start: Data Analysis Need
What type of output do you need?
Discrete Categories
Classification
Use Cases
Categorical Yield Predictions
Risk Category Assessment
General Planning
Continuous Values
Regression
Use Cases
Precise Yield Estimates
Financial Projections
Resource Planning

Implementation Strategy

Data Requirements:

Weather data

Soil information

Agricultural practices

Model Deployment:

Regular retraining

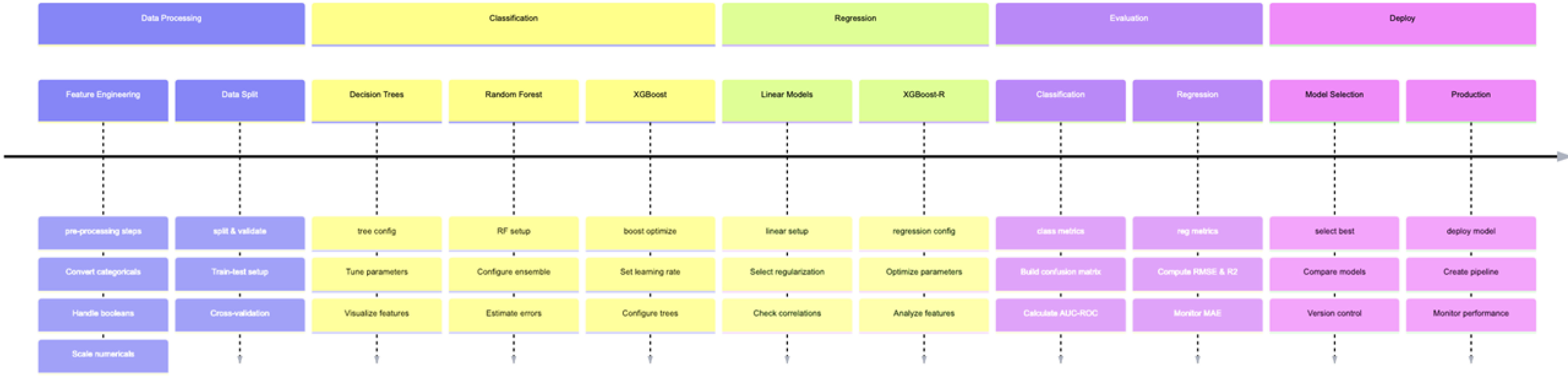
Performance monitoring

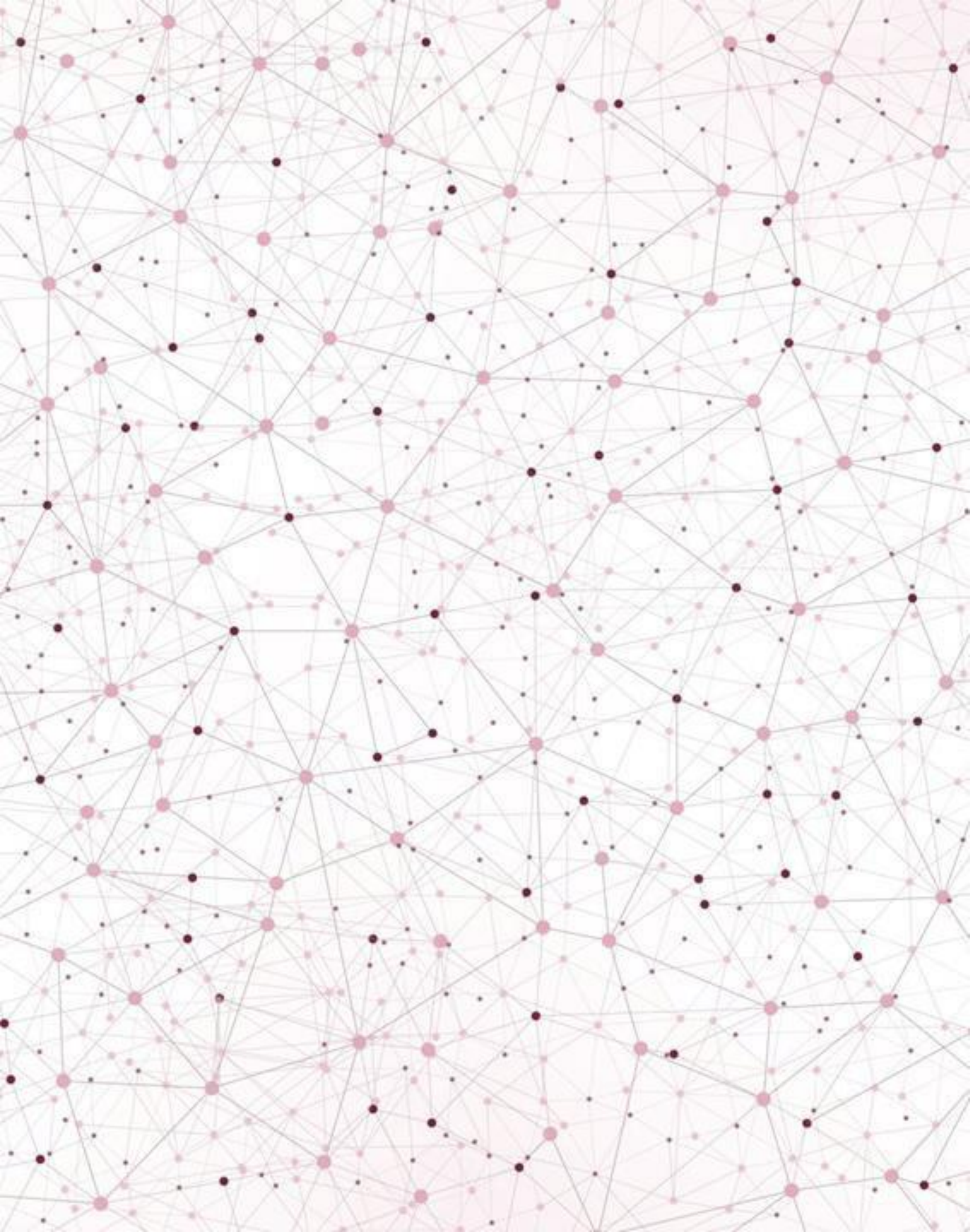
Integration Points:

Farm management systems

Financial planning tools [Image Type: Implementation roadmap(if possible need to make it)]

ML Model Implementation Roadmap

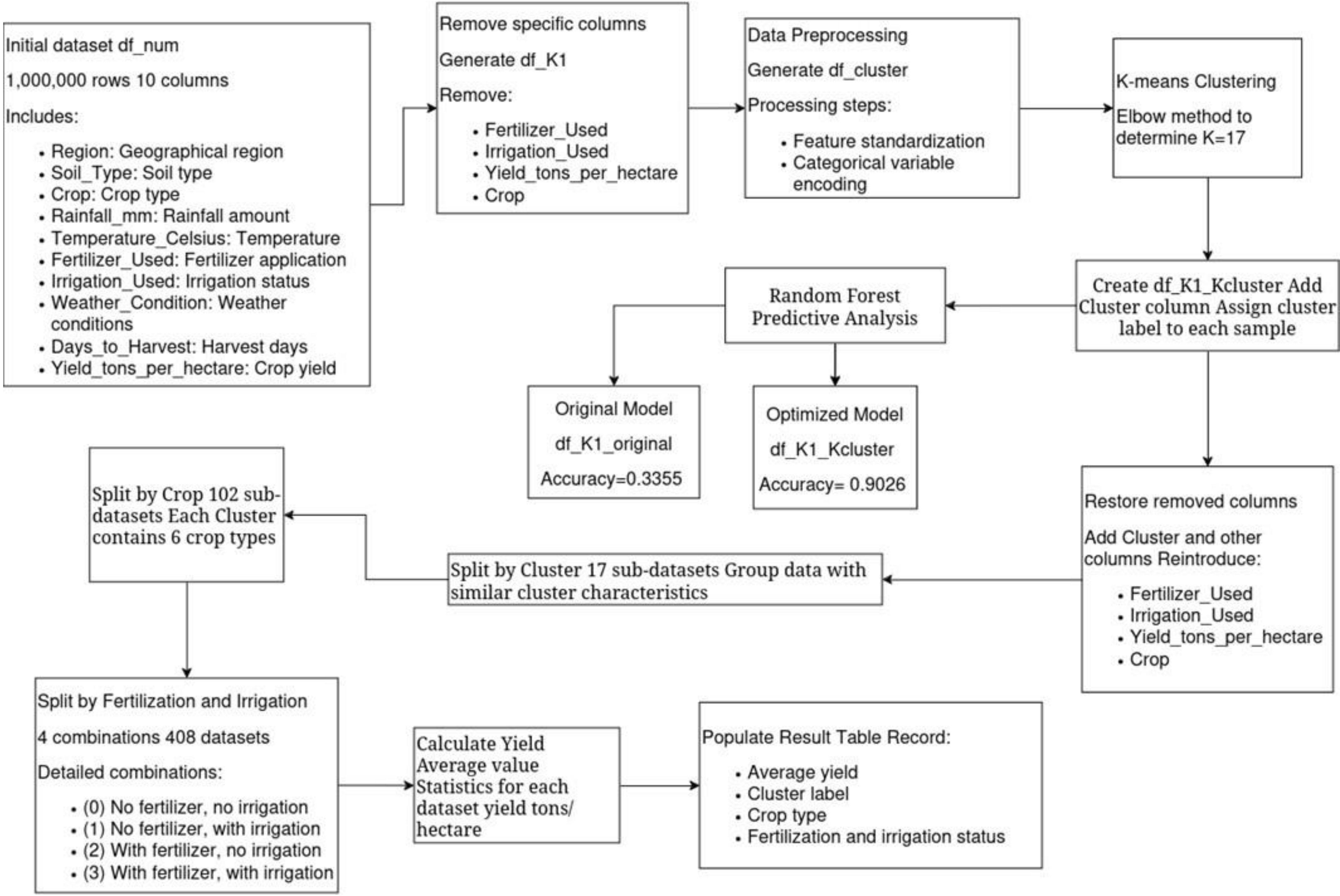




Decision Support on Irrigation and Fertilization

Objective: Determine whether fertilization or irrigation is needed in different environment and local crop growing costs.

Basic Idea


Initial dataset df_num
1,000,000 rows 10 columns
Includes:
• Region: Geographical region
• Soil_Type: Soil type
• Crop: Crop type
• Rainfall_mm: Rainfall amount
• Temperature_Celsius: Temperature
• Fertilizer_Used: Fertilizer application
• Irrigation_Used: Irrigation status
• Weather_Condition: Weather conditions
• Days_to_Harvest: Harvest days
• Yield_tons_per_hectare: Crop yield
Remove specific columns
Generate df_K1
Remove:
• Fertilizer_Used
• Irrigation_Used
• Yield_tons_per_hectare
• Crop
Data Preprocessing
Generate df_cluster
Processing steps:
• Feature standardization
• Categorical variable encoding
K-means Clustering
Elbow method to determine K=17
Create df_K1_Kcluster Add Cluster column Assign cluster label to each sample
Random Forest Predictive Analysis
Original Model
df_K1_original
Accuracy=0.3355
Optimized Model
df_K1_Kcluster
Accuracy= 0.9026
Restore removed columns
Add Cluster and other columns Reintroduce:
• Fertilizer_Used
• Irrigation_Used
• Yield_tons_per_hectare
• Crop
Split by Cluster 17 sub-datasets Group data with similar cluster characteristics
Split by Crop 102 sub-datasets Each Cluster contains 6 crop types
Split by Fertilization and Irrigation
4 combinations 408 datasets
Detailed combinations:
• (0) No fertilizer, no irrigation
• (1) No fertilizer, with irrigation
• (2) With fertilizer, no irrigation
• (3) With fertilizer, with irrigation
Calculate Yield Average value Statistics for each dataset yield tons/hectare
Populate Result Table Record:
• Average yield
• Cluster label
• Crop type
• Fertilization and irrigation status

Result

	Cluster	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Corps	F&I	Yield Tons Per Hectare																
Rice	0	3.2959	3.3274	3.3134	3.2759	3.3133	3.3466	3.3213	3.2915	3.2606	3.2887	3.2870	3.3410	3.3755	3.2912	3.3086	3.2981	3.2884
	1	4.4829	4.5279	4.4424	4.5243	4.4789	4.5387	4.5286	4.5069	4.5146	4.4644	4.5029	4.5038	4.5666	4.5286	4.4646	4.5247	4.5464
	2	4.7669	4.8002	4.7866	4.8477	4.8075	4.8027	4.8032	4.8104	4.7909	4.7947	4.8110	4.8110	4.8137	4.8196	4.7971	4.8192	4.7672
	3	6.0025	6.0194	6.0384	5.9963	5.9572	5.9853	6.0249	5.9817	5.9867	6.0205	6.0100	6.0132	6.0682	6.0380	6.0006	5.9947	5.9500
Barley	0	3.3162	3.2592	3.2893	3.2749	3.2904	3.2713	3.3027	3.3621	3.3471	3.2952	3.4100	3.3242	3.3178	3.2420	3.3004	3.3142	3.3326
	1	4.4771	4.5034	4.4637	4.4227	4.5481	4.4612	4.4725	4.4999	4.4790	4.5057	4.4895	4.4544	4.4780	4.4843	4.4674	4.5257	4.4909
	2	4.8369	4.7787	4.7653	4.7845	4.8771	4.7793	4.7649	4.8300	4.8263	4.8199	4.7849	4.8389	4.7456	4.8124	4.8195	4.8178	4.8052
	3	6.0506	6.0127	5.9878	5.9971	6.0596	5.9398	6.0248	6.0181	6.0018	5.9977	6.0194	6.0395	5.9647	6.0138	6.0097	5.9913	6.0161
Soy bean	0	3.2773	3.3255	3.2807	3.3585	3.2970	3.3248	3.3588	3.3215	3.2693	3.2932	3.2464	3.3083	3.3636	3.3110	3.3483	3.3100	3.2290
	1	4.4960	4.5189	4.4519	4.5556	4.5326	4.4914	4.5183	4.5331	4.4857	4.4922	4.5229	4.5113	4.5350	4.4943	4.5008	4.4832	4.5051
	2	4.7839	4.7862	4.7748	4.7920	4.8588	4.7538	4.7873	4.8018	4.7875	4.7908	4.7459	4.7716	4.8415	4.8219	4.8100	4.7684	4.8072
	3	6.0000	6.0168	6.0191	6.0373	5.9271	6.0523	6.0196	5.9917	6.0332	6.0191	5.9940	6.0272	6.0313	5.9978	5.9718	5.9607	6.0252
Wheat	0	3.2602	3.3412	3.2937	3.3237	3.3234	3.3512	3.3280	3.3242	3.2775	3.3149	3.3053	3.2528	3.2634	3.3091	3.3096	3.3287	3.2913
	1	4.4639	4.4911	4.5295	4.4662	4.5255	4.4787	4.4379	4.5567	4.4643	4.5069	4.5194	4.4592	4.5047	4.5021	4.4900	4.5054	4.4866
	2	4.7697	4.7631	4.8251	4.8195	4.8024	4.8276	4.7891	4.8455	4.8107	4.7806	4.8360	4.8511	4.8188	4.8275	4.8266	4.7844	4.7754
	3	6.0376	5.9931	6.0359	5.9903	5.9904	5.9976	5.9877	6.0006	6.0138	5.9923	5.9660	5.9613	5.9470	5.9889	5.9230	5.9530	6.0267
Maize	0	3.3030	3.2997	3.3340	3.2818	3.2589	3.2796	3.2712	3.3457	3.3282	3.3119	3.2716	3.3138	3.3077	3.3050	3.2922	3.2788	3.2721
	1	4.5067	4.4709	4.5628	4.5251	4.5006	4.4667	4.4802	4.5127	4.4702	4.4781	4.5234	4.5601	4.4791	4.5266	4.4941	4.4885	4.4725
	2	4.8001	4.7855	4.7756	4.7785	4.8410	4.7561	4.7848	4.8002	4.8685	4.7980	4.7488	4.7747	4.8435	4.7256	4.7916	4.7337	4.8554
	3	5.9884	6.0174	6.0010	6.0209	5.9848	5.9834	6.0150	5.9881	5.9736	5.9992	6.0102	5.9832	5.9880	5.9907	5.9554	5.9398	5.9805
Cotton	0	3.2937	3.3149	3.3033	3.3211	3.2473	3.3424	3.3051	3.3339	3.3618	3.2559	3.3160	3.2777	3.2458	3.3203	3.3093	3.3443	3.2810
	1	4.5174	4.5195	4.5355	4.4993	4.4763	4.5265	4.5042	4.5127	4.4998	4.4800	4.4673	4.4795	4.5333	4.5290	4.5321	4.4363	4.4743
	2	4.7660	4.8167	4.7051	4.8174	4.8389	4.8515	4.8257	4.8634	4.7513	4.8108	4.8094	4.7469	4.7896	4.7947	4.7474	4.8736	4.7875
	3	5.9849	5.9532	5.9889	5.9459	5.9879	6.0318	6.0491	5.9879	5.9852	5.9862	5.9637	6.0375	5.9887	5.9703	6.0525	5.9787	6.0180

Real-World Application

There is a farmer who wants to grow rice and the environment of his fiel belongs to cluster 0.

	Fertilizer	Irrigation
Cost	$10	$15

	Rice
Price	$10

	Cluster	0	
Corps	F&I		Income
Rice	0	3.2959	$32.96
	1	4.4829	$29.83
	2	4.7669	$37.67
	3	6.0025	$35.03

So, the farmer should only fertilize the rice to get the most Income.

Thank You