**Agriculture Data analysis:**

**Problem statement:**

**The dataset includes the following columns:**

Domain Code: Represents a code identifying the domain of the data, such as a specific sector, category.

Domain: A descriptive name for the domain associated with the Domain Code.

Area Code (M49) An identifier for geographic areas, likely based on the M49 standard (used by the United Nations).

Area The name of the geographic area corresponding to the Area Code (M49).

Element Code A numeric code representing specific data elements

Element A descriptive label for the data element associated with the Element Code.

Item Code (CPC) A code identifying specific items or products.

Item The name of the product or item

Year Code A code for the year the data pertains to

Year The calendar year of the data.

Unit The unit of measurement for the data in the Value column.

Value The numeric value or observation corresponding to the specified domain, area, element, item, and year.

Flag A code indicating the quality, status, or source of the data.

Flag Description A descriptive explanation of the Flag.

Note Additional comments or explanations for certain rows

**The project aims to leverage historical data to provide predictive insights into agricultural values.**

1. Data Preprocessing

Performed extensive data preprocessing to ensure the dataset's quality for analysis and forecasting. This involved the following steps:

* Handling Missing Values:
  + identified missing values in key columns and addressed them using statistical aggregators. For numerical fields, used mean or median imputation based on the data distribution. For categorical variables, applied mode imputation or created an "Unknown" category.
* Encoding Categorical Variables:
  + converted categorical features into numerical format using one-hot / label encoding for non-ordinal (nominal) data and ordinal encoding for ordinal features to facilitate model compatibility.
* Scaling and Normalization:
  + normalized continuous variables such as "Value" using Min-Max scaling for models sensitive to feature magnitude (e.g., Ridge and Lasso Regression) to ensure better convergence and performance.

2. Exploratory Data Analysis Project (EDA)

Conducted a comprehensive exploratory analysis to understand feature distributions, relationships, and potential insights:

1. Categorical Data Distribution:
   * used seaborn's countplot() function to visualize the distribution of categorical variables (e.g., "Domain," "Element Code," "Element," "Year").
2. Continuous Data Distribution:
   * used seaborn's boxplot() function to identify outliers and understand the spread of numerical variables.
3. Feature Relationships:
   * utilized pairplots to inspect relationships between numeric features.
4. Correlation Analysis:
   * generated a heatmap to identify correlations between variables, revealing a weak positive correlation between "Area Code" and "Element Code" and a slight negative correlation between "Year" and "Value."
5. Trend Analysis:
   * plotted line charts to analyse how agricultural values evolved over time, noting a general upward trend despite minor fluctuations.

### **Key Visualizations and Outcomes**

|  |  |  |
| --- | --- | --- |
| KPI | Visualization | Outcome |
| Features vs. Value | Boxplot | Identified outliers across variables like "Element". |
| Categorical Data Distribution | Countplot | Displayed frequency distributions of key attributes. |
| Yearly Trends in Agricultural Value | Line plot | Observed a peak in 2021 for production values. |
| Production by Area | Boxplot | India leads in production, followed by China. |
| Production Value by Year | Bar plot | Confirmed 2021 as the year with the highest output. |
| Relationships Between Features | Pairplot | Explored dependencies across numeric variables. |

Applying Machine Learning Models

3. Model Selection

Implemented various regression models to predict agricultural values, beginning with simpler algorithms to establish baseline performance:

* Linear Regression: model for initial benchmarks.
* Lasso & Ridge Regression: Addressed multicollinearity and improved generalization.
* Decision Tree Regressor: Captured non-linear relationships effectively.
* Random Forest Regressor: Enhanced performance using ensemble learning.
* Gradient Boost Regressor: Sequential learning model to correct errors.
* XGBoost Regressor: Optimized for speed and performance in handling complex patterns.

4. Model Evaluation

Have split the dataset into 75% training and 25% test data and evaluated models using:

* Mean Squared Error (MSE): Quantified average squared prediction errors.
* R-squared (R²): Assessed how well models explained data variance.

**Default models:**

* Decision Tree Regressor: High accuracy (0.9327) and R2 score with minimal overfitting.
* Random Forest Regressor: Highest accuracy (0.9910) and low overfitting.
* XGBoost Regressor: Good balance between train and test scores with an R2 score of 0.8019.
* Gradient Boost Regressor: Lower performance with accuracy around 0.3305.
* Linear, Lasso, and Ridge Regressions: Very low accuracy scores and significant MSE scores.

**Hyperparameter Tuning**

* Decision Tree Regressor: Maintained high accuracy.
* Random Forest Regressor: Optimal parameters improved overfitting issues.
* XGBoost Regressor: Reduced performance with parameter tuning.
* Gradient Boost Regressor: Improved performance with optimal parameters.
* Lasso and Ridge Regressions: Minor improvements with tuning.

**Best Model:**

The Random Forest Regressor appears to be the best model, especially when using default parameters. It shows the highest accuracy (0.9910) with low overfitting.

When hyperparameters are tuned, the Random Forest Regressor still performs very well, indicating its robustness and stability.

From MSE & R2 metrics, the Random Forest Regressor stands out for its low MSE and high R2 score, indicating both its accuracy and robustness in predictions. The Decision Tree Regressor also performs quite well, while models like Gradient Boost Regressor show significant improvement with hyperparameter tuning.

**Comparing Model Accuracy and Tuning**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Defaults parameters | | | |
| models | Train score | Overfitting there or not | R2 | MSE score | |
| Decision tree regressor | 1 | 6.73% | 0.932693222 | 4.3417E+12 | |
| Random forest regressor | 0.9969677 | 0.60% | 0.990981444 | 5.81752E+11 | |
| Extreme gradient boost regressor (Xgboost regressor ) | 0.9782286 | 1.05% | 0.801914169 | 2.08335E+12 | |
| Gradient boost regressor | 0.2947 | -3.58% | 0.330456182 | 4.31896E+13 | |
| linear regression | 0.0119652 | 0.08% | 0.011127885 | 6.37882E+13 | |
| Lasso regression | 0.0121264 | 0.08% | 0.011346201 | 6.37742E+13 | |
| ridge regression | 0.0121264 | 0.08% | 0.011346257 | 6.37742E+13 | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Hyperparameter tuning | | | | | |
| models | Best parameters | R2 score | Train score | Overfitting there or not | R2 | MSE score |  |
| Decision tree regressor | - | 0.93269 | 1 | 6.73% | 0.9327 | 4.3421E+12 |  |
| Random forest regressor | {'max\_depth': 9, 'n\_estimators': 15} | 0.91182 | 0.9008019 | -1.10% | 2384978.11 | 5.6881E+12 |  |
| Extreme gradient boost regressor (Xgboost regressor ) | {'subsample': 1.0, 'reg\_lambda': 1, 'reg\_alpha': 4, 'n\_estimators': 20, 'min\_child\_weight': 3, 'max\_depth': 3, 'learning\_rate': 0.5, 'colsample\_bytree': 1.0} | 0.56001 | 0.4673381 | -9.27% | 0.56000776 | 2.8382E+13 |  |
| Gradient boost regressor | {'learning\_rate': 0.1,'loss': 'squared\_error','max\_depth': 11,'n\_estimators': 9} | 0.8019 | 0.8297817 | 2.79% | 0.80189615 | 1.2779E+13 |  |
| linear regression | - | - | - | - | - | - |  |
| Lasso regression | {'max\_depth': 2, 'n\_estimators': 2} | 0.0114 | 0.0121 | 0.07% | 0.01138376 | 6.3772E+13 |  |
| ridge regression | {'criterion': 'gini', 'max\_depth': 5, 'min\_samples\_leaf': 1, 'min\_samples\_split': 5} | 0.01135 | 0.0121264 | 0.08% | 0.0113462 | 6.3774E+13 |  |