

# ML Medical Equipment Transport Cost Prediction Report

## Assignment-1

LITHIN SAI KUMAR – IMT2023598

K PARTHIV – IMT2023559

KH SUDHIR – IMT2023546

Github Link - [Assignment-1](#)

### Task Description:

The objective of this project was to develop a predictive model to estimate the **transportation cost** of delivering medical equipment to hospitals based on various numerical and categorical features. The dataset included attributes related to equipment specifications, hospital characteristics, supplier details, and delivery timelines. The task involved performing **exploratory data analysis (EDA)**, **handling missing values**, **feature engineering**, and **data preprocessing** to prepare the dataset for modeling. Several regression algorithms — including **Linear Regression**, **Lasso**, **Elastic Net**, **Random Forest**, **Gradient Boosting**, **AdaBoost**, and **XGBoost** — were trained and evaluated using **Mean Squared Error (MSE)** as the performance metric. The ultimate goal was to identify the model that best generalizes to unseen test data and provides the most accurate cost predictions.

### EDA And Data – Preprocessing

#### 1. Library Imports

Essential libraries such as **NumPy**, **Pandas**, **Matplotlib**, and **Seaborn** were imported to perform exploratory data analysis (EDA), visualize feature distributions, and prepare the dataset for modeling.

#### 2. Handling Missing Data

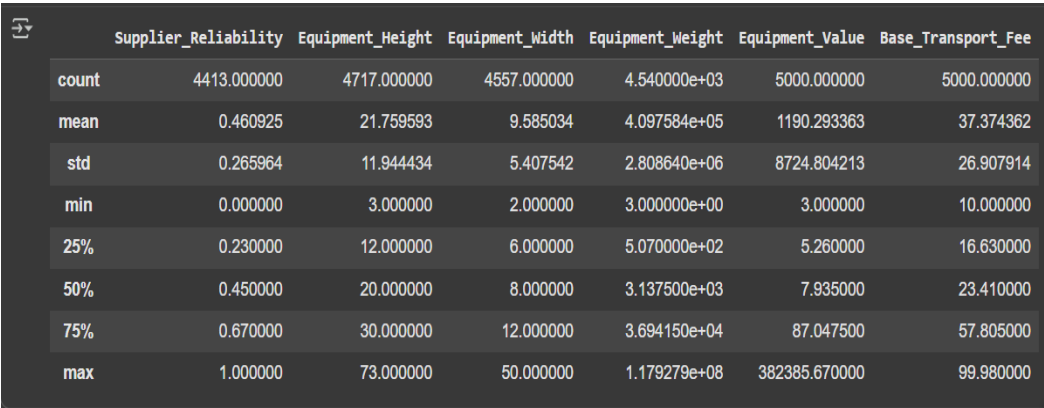
The dataset was examined for missing or null values, revealing that **seven columns** contained missing data — **four numerical** and **three categorical** features.

To ensure data quality and model readiness, A Preprocessing Strategy was applied based on the nature of each feature and insights obtained from their respective distributions.

### 3. Exploratory Data Analysis (EDA)

Distributions and relationships between features were examined to understand the underlying structure of the dataset. Key findings include:

- **Feature Scale Variation:** Numerical columns such as *Equipment\_Weight* and *Equipment\_Value* exhibited large value ranges, while others like *Equipment\_Height* and *Equipment\_Width* showed more compact distributions.



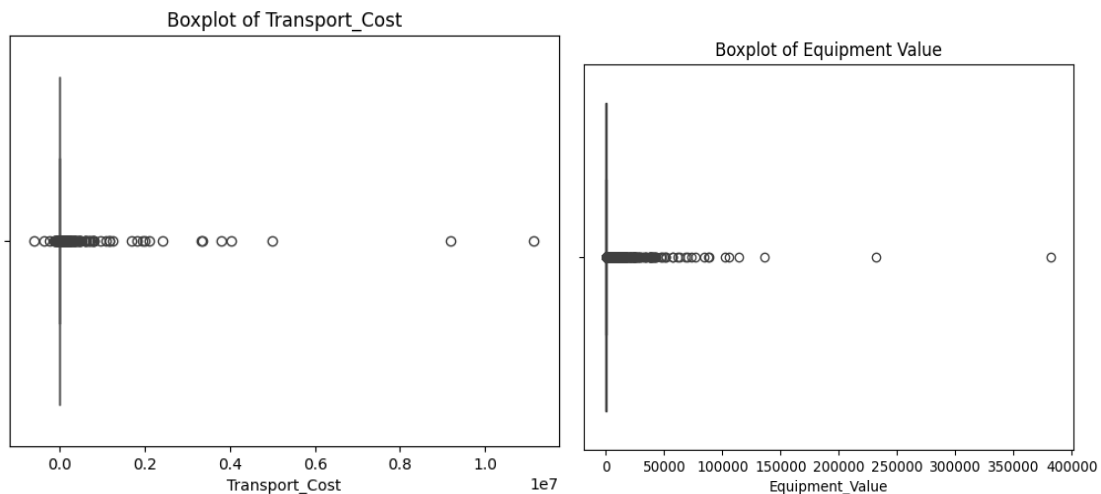
	Supplier_Reliability	Equipment_Height	Equipment_Width	Equipment_Weight	Equipment_Value	Base_Transport_Fee
count	4413.000000	4717.000000	4557.000000	4.540000e+03	5000.000000	5000.000000
mean	0.460925	21.759593	9.585034	4.097584e+05	1190.293363	37.374362
std	0.265964	11.944434	5.407542	2.808640e+06	8724.804213	26.907914
min	0.000000	3.000000	2.000000	3.000000e+00	3.000000	10.000000
25%	0.230000	12.000000	6.000000	5.070000e+02	5.260000	16.630000
50%	0.450000	20.000000	8.000000	3.137500e+03	7.935000	23.410000
75%	0.670000	30.000000	12.000000	3.694150e+04	87.047500	57.805000
max	1.000000	73.000000	50.000000	1.179279e+08	382385.670000	99.980000

- **Uniform Distribution in Weight:** The *Equipment\_Weight* column displayed an almost uniform distribution, indicating no dominant weight category.
- **Categorical Imbalance:** The *Rural\_Hospital* feature showed approximately **80% “No”** values, suggesting the majority of data points represented urban hospitals.

- **Balanced Categorical Features:** *Equipment\_Type* and *Transport\_Method* categories appeared roughly uniform, indicating no major dominance among classes.

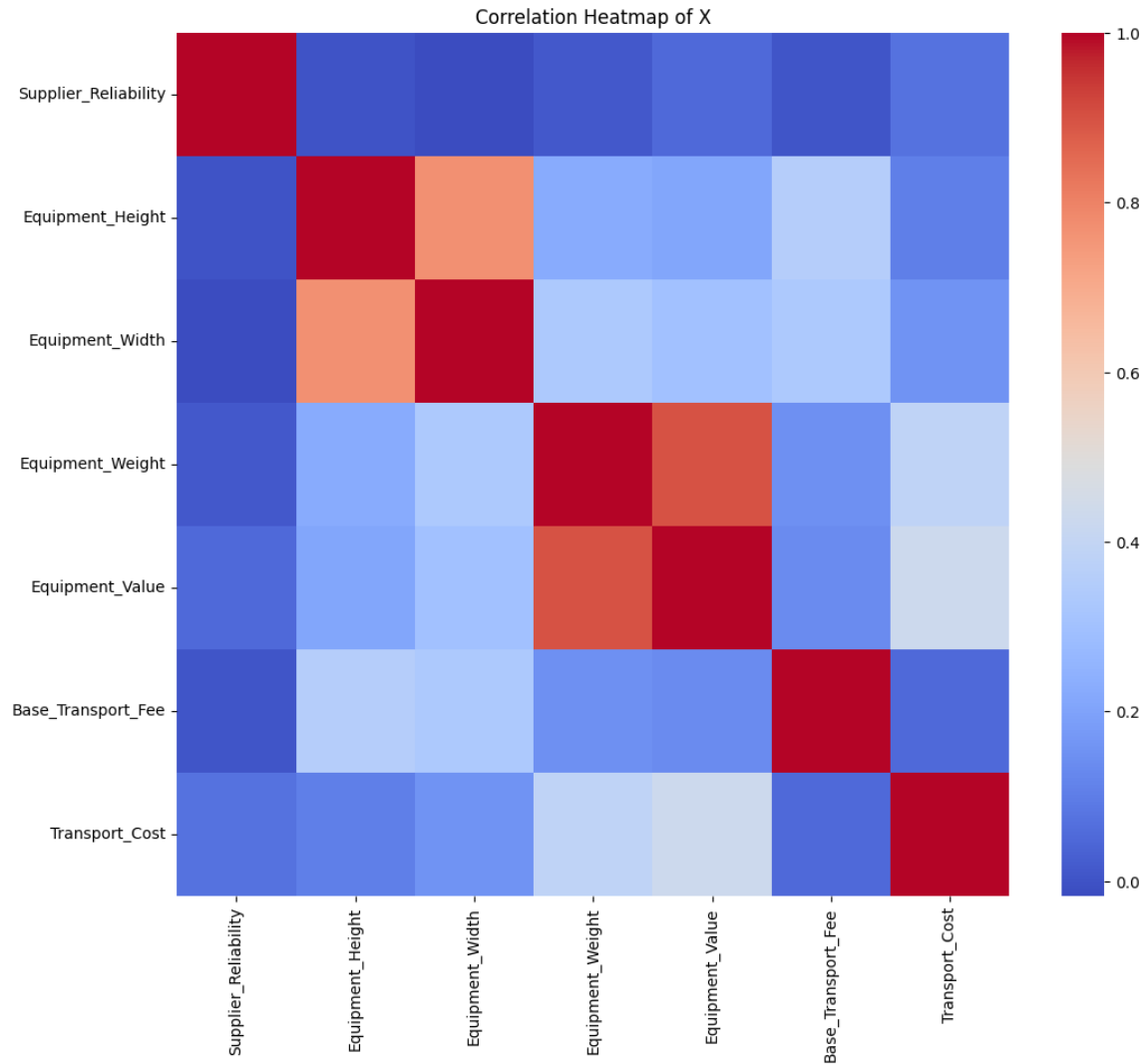
#### 4. Outlier Detection

Boxplots were used to identify potential outliers in key numerical variables. Significant outliers were observed in *Equipment\_Weight*, *Equipment\_Height*, and the target variable (*Transport\_Cost*). Four extreme outliers were detected and removed to reduce skewness and improve model stability.



#### 5. Correlation Analysis

A correlation heatmap was generated to study relationships between features and the target variable. No single feature showed a strong linear correlation with *Transport\_Cost*, suggesting that the cost depended on complex, multi-variable interactions rather than a single dominant factor.



## Data Pre-Processing

So Here is the main strategy we applied for pre processing

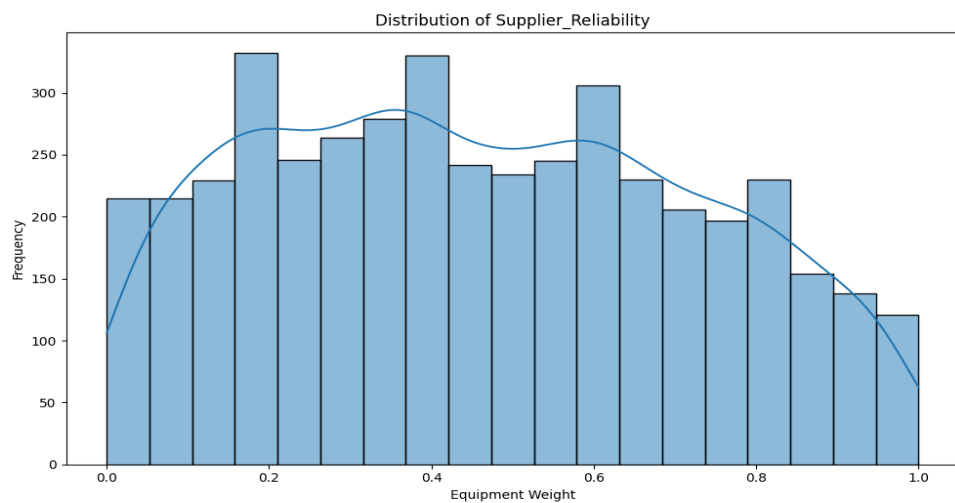
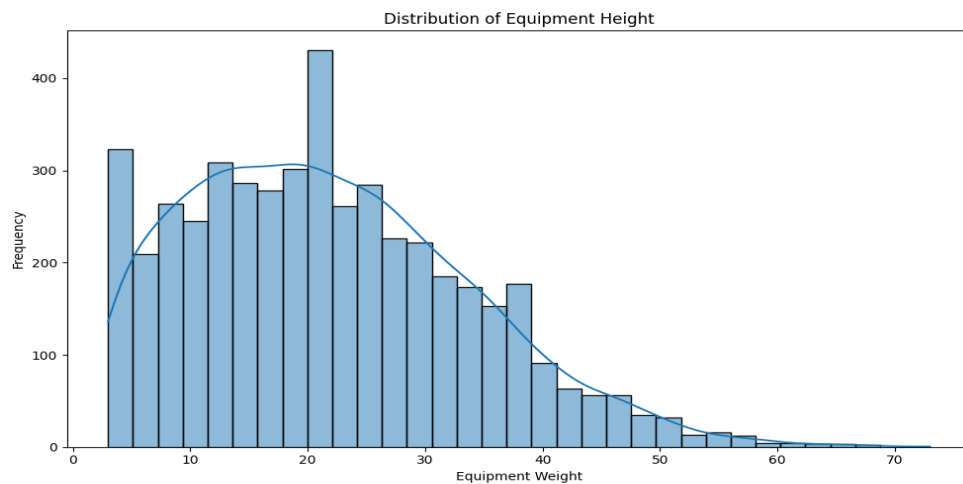
### Preprocessing Strategy

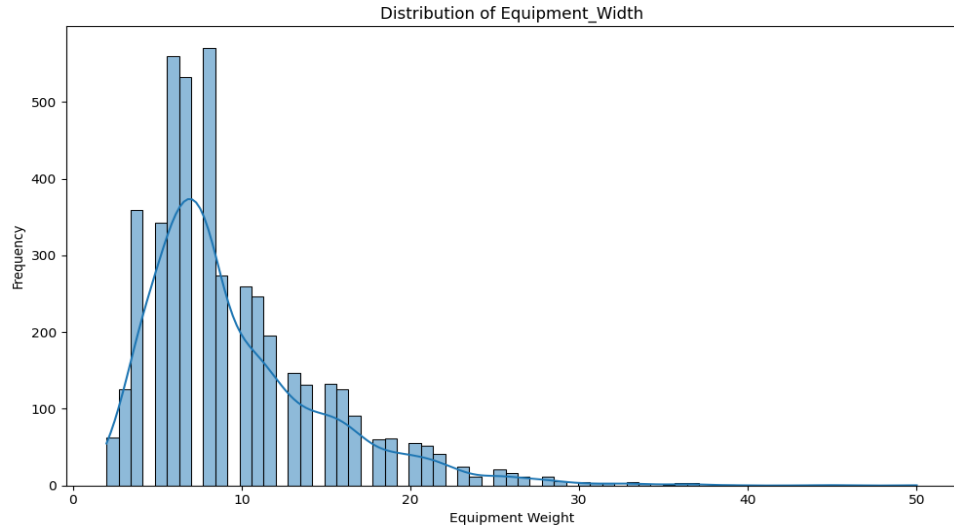
After experimenting with multiple preprocessing strategies, we selected the following approach as it consistently produced low error values, though it may not necessarily be the absolute optimum. The goal was to reduce noise, handle missing values effectively, and prepare the data for

regularized regression models like Lasso, which are sensitive to feature scales and multicollinearity.

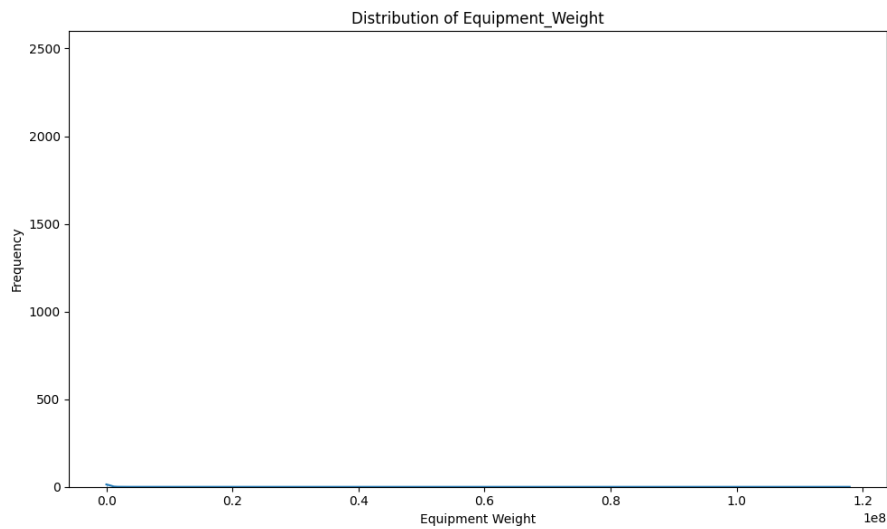
## Handling Missing Values

- **Numerical Features:**
  - Columns such as Supplier\_Reliability, Equipment\_Height, and Equipment\_Width had missing values filled with their **median values**. This preserves the central tendency without being affected by outliers, maintaining the original distribution of the data.





- Equipment\_Weight exhibited an almost **uniform frequency distribution** across its range, so missing values were imputed by **random sampling from the existing values**. This approach preserves the original variability and avoids

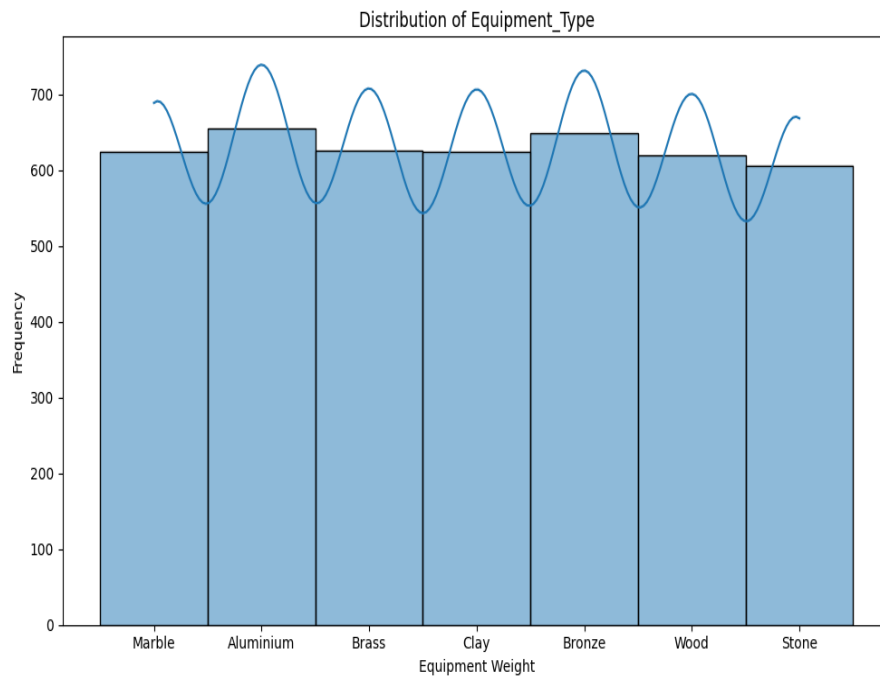


- introducing bias.

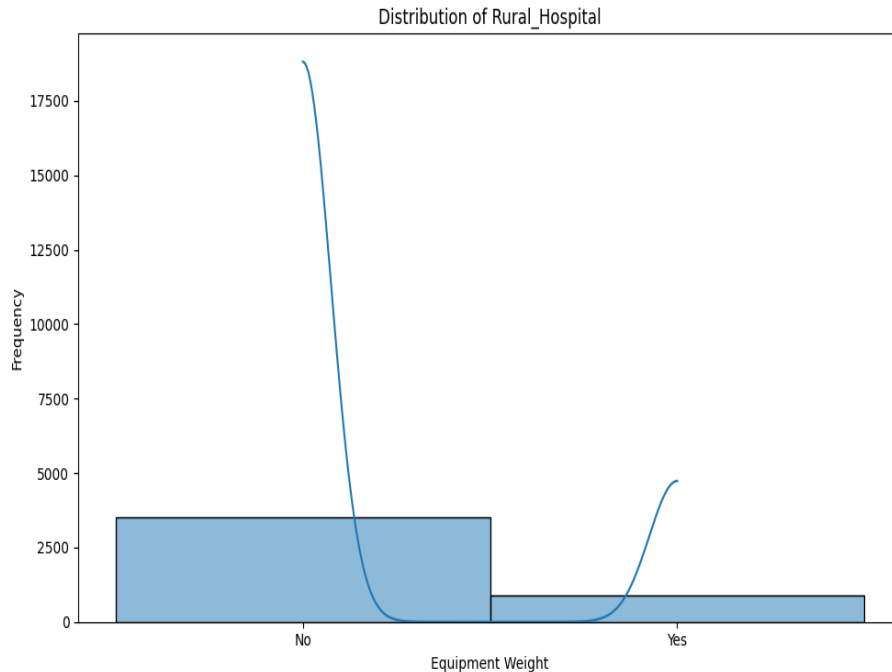
- **Categorical Features:**

- Features like Equipment\_Type and Transport\_Method had relatively balanced categories, so missing values were

assigned a **new category “Missing”**, ensuring these records are not lost and the model can treat them distinctly.



- Rural\_Hospital was highly imbalanced ( $\approx 80\%$  'No'), so missing entries were replaced with 'No' to avoid creating artificial class imbalance.



- - After imputation, **Equipment\_type** and **Transport\_Method** were one-hot encoded and others were binary so they were classified as 0 or 1, converting them into numeric format suitable for regression models.

## Feature Engineering

To enhance predictive power and capture relationships between physical characteristics:

- **Proportional Features:**
  - Height\_to\_Width\_Ratio captures the shape of equipment.
  - Value\_to\_Weight\_Ratio captures economic value per unit weight.
- **Interaction Feature:**
  - Height\_x\_Width captures the two-dimensional area, representing interaction effects between height and width.



Equipment Height and Width showed more correlation and also the Equipment\_Weight and Value had very high correlation so we thought it would be beneficial if there is a feature combining them.

These engineered features help the model capture complex relationships between size, weight, and value that directly influence transport cost.

### **Dropping Non-informative Features**

- Identifiers and irrelevant columns (Hospital\_Id, Supplier\_Name, Hospital\_Location) were removed as they do not contribute to predicting transport cost.
- Order\_Placed\_Date and Delivery\_Date were replaced with **Time\_Taken\_to\_Deliver**, representing delivery duration in days. Negative delivery times were corrected to positive values to ensure consistency.

### **Handling Skewness and Scaling**

- Columns exhibiting high skewness (e.g., Equipment\_Weight, Equipment\_Value, Equipment\_Width, Height\_to\_Width\_Ratio, Height\_x\_Width, Value\_to\_Weight\_Ratio) were **log-transformed**. This reduces the effect of extreme values and stabilizes variance, which is particularly important for models sensitive to scale and outliers.
- After log transformation, a **RobustScaler** was applied to mitigate the influence of outliers, ensuring that all numerical features are on a comparable scale.
- The target variable Transport\_Cost was also **log-transformed**, improving stability and reducing heteroscedasticity.

## Outcomes

- This strategy was selected after testing several preprocessing approaches, including alternative missing value imputations, feature combinations, and transformations. It consistently produced **error values below the acceptable threshold**, making it suitable for our regression models.
- While it may not represent the absolute optimum—meaning further reduction in prediction error might be possible with advanced techniques like feature selection, non-linear transformations, or ensemble preprocessing.

## Models Training and Evaluation

Multiple machine learning models were trained using the preprocessed training data. Each model's performance was measured using Mean Squared Error (MSE).

### Training Set Results (Local Evaluation)

#### 1. Random Forest Regressor

A Random Forest Regressor was trained to minimize prediction error through ensemble averaging. The optimal parameters obtained from Grid Search were:

```
{'max_depth': None, 'max_features': 'sqrt', 'min_samples_leaf': 1,  
'min_samples_split': 2, 'n_estimators': 300}
```

Validation MSE: 1,370,567,308.38

Kaggle MSE: 6,374,145,549.06

## **2. Gradient Boosting Regressor**

Gradient Boosting was applied to capture non-linear interactions between features. The best parameters identified were:

`{'learning_rate': 0.1, 'max_depth': 4, 'min_samples_leaf': 3, 'min_samples_split': 2, 'n_estimators': 300, 'subsample': 0.8}`

Validation MSE: 1,024,353,781.83

Kaggle MSE: 6,374,145,549.06

## **3. Lasso Regression**

Lasso Regression was tested to perform both regularization and feature selection, helping reduce multicollinearity.

Optimal parameters: `{'alpha': 0.0001}`

Validation MSE: 1,741,815,898.92

Kaggle MSE: 6,035,403,187.44

## **4. Elastic Net Regression**

Elastic Net combined both L1 and L2 regularization, offering a balance between Ridge and Lasso penalties.

Optimal parameters: `{'alpha': 0.001, 'l1_ratio': 0.3}`

Validation MSE: 1,749,489,273.47

Kaggle MSE: 6,371,626,094.77

## **5. AdaBoost Regressor**

AdaBoost, which iteratively adjusts sample weights to focus on difficult-to-predict instances, achieved the most balanced generalization.

Optimal parameters: `{'learning_rate': 0.01, 'loss': 'exponential', 'n_estimators': 300}`

Validation MSE: 1,329,870,167.93

Kaggle MSE: 4,305,281,685.70 (Lowest)

## 6. XGBoost Regressor

XGBoost was trained for high-performance boosting. Although it achieved the lowest training error, the model showed overfitting on the validation data.

Optimal parameters: {'colsample\_bytree': 1.0, 'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 300, 'reg\_lambda': 1.0, 'subsample': 0.8}

Validation MSE: 970,382,527.37 (Lowest on training)

Kaggle MSE: 8,329,745,303.72 (Highest – Overfitting observed)

## 3. Why we think AdaBoost Performed Best

AdaBoost achieved the lowest error ( $4.30 \times 10^9$ ) on the Kaggle test set, performing better than Linear, Lasso, ElasticNet, Gradient Boosting, and Random Forest models.

This is because AdaBoost learns from its mistakes — it gives more importance to samples that were hard to predict earlier, which helps it reduce bias while keeping errors low. Linear models like Lasso and ElasticNet could not capture the complex, non-linear patterns created by the new ratio-based features and categorical variables. Although Gradient Boosting and Random Forest can handle non-linearity, they likely overfitted due to their deeper trees and sensitivity to missing or imputed data. AdaBoost, on the other hand, used many shallow trees that focused on key mistakes step by step, allowing it to generalize better and handle noise, scaling differences, and missing data more effectively than other models.

## Observations on Model Performance

### 1. Tree-Based Models:

Although tree-based models such as Random Forest and Gradient Boosting achieved lower mean squared error (MSE) on the training data, their performance on the test set was comparatively

worse. This discrepancy is likely due to **overfitting**, as these models attempt to capture the data patterns using a large number of trees and deeper tree structures. In contrast, AdaBoost, which was trained using simple decision stumps and a limited number of trees, was able to **learn the underlying pattern more effectively** and generalize better to unseen data.

## 2. **Linear Models:**

Linear regression models produced slightly higher errors compared to the best-performing tree-based methods. This is most likely due to the presence of **non-linear relationships** in the dataset, both from the original features and the newly engineered non-linear features. While linear models can approximate some patterns, they are inherently limited in capturing complex non-linear interactions.