Predicting Resource Allocation Efficiency in Lean Construction Projects using Machine Learning

📌 Objective

To build a predictive model for **Resource Allocation Efficiency** in Lean Construction projects using machine learning techniques. This project supports data-driven decision-making in optimizing the allocation of labor, equipment, and materials.

💼 Tools & Technologies

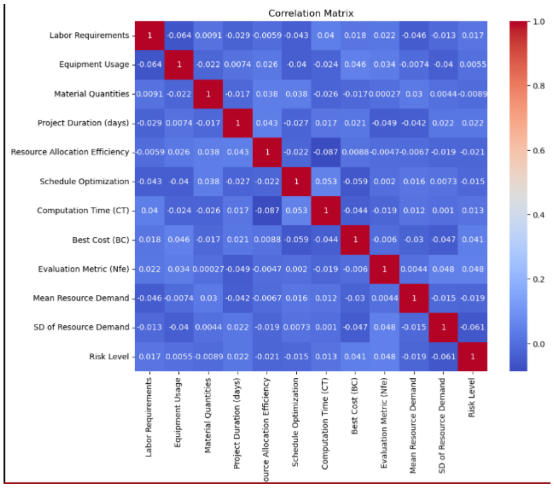
- Python, Pandas, Scikit-Learn, XGBoost, Matplotlib, Seaborn
- Jupyter Notebook
- Dataset: Construction_Dataset.csv (Lean Construction project simulation)

🛠 Phase Breakdown

📊 Phase 1: Exploratory Data Analysis (EDA)

- Performed visual analysis using heatmaps and scatterplot matrices to explore correlations between features and the target.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 12 columns):
 #   Column                         Non-Null Count  Dtype
---  ------                         --------------  -----
 0   Labor Requirements             1000 non-null   int64
 1   Equipment Usage                1000 non-null   int64
 2   Material Quantities            1000 non-null   float64
 3   Project Duration (days)        1000 non-null   int64
 4   Resource Allocation Efficiency 1000 non-null   float64
 5   Schedule Optimization          1000 non-null   int64
 6   Computation Time (CT)          1000 non-null   float64
 7   Best Cost (BC)                 1000 non-null   float64
 8   Evaluation Metric (Nfe)        1000 non-null   int64
 9   Mean Resource Demand           1000 non-null   float64
 10  SD of Resource Demand          1000 non-null   float64
 11  Risk Level                     1000 non-null   int64
dtypes: float64(6), int64(6)
memory usage: 93.9 KB
```




```
Resource Allocation Efficiency     1.000000
Project Duration (days)            0.042610
Material Quantities                0.037555
Equipment Usage                    0.025672
Best Cost (BC)                     0.008783
Evaluation Metric (Nfe)           -0.004725
Labor Requirements                -0.005866
Mean Resource Demand              -0.006678
SD of Resource Demand             -0.019093
Risk Level                        -0.021299
Schedule Optimization             -0.021800
Computation Time (CT)             -0.087012
Name: Resource Allocation Efficiency, dtype: float64
```

- Identified generally weak linear relationships among most variables.

Phase 2: Data Preprocessing

- Applied One-Hot Encoding to categorical variables (Risk Level).
- Standardized numerical features using StandardScaler() to ensure fair comparison across models.

```python
from sklearn.preprocessing import StandardScaler
import pandas as pd

# Salin data untuk menjaga keutuhan data asli
df_scaled = df.copy()

# One-Hot Encoding untuk kolom kategorikal
df_scaled = pd.get_dummies(df_scaled, columns=['Risk Level'], drop_first=True)

# Pisahkan fitur dan target
target = df_scaled['Resource Allocation Efficiency']
features = df_scaled.drop(columns=['Resource Allocation Efficiency'])

# Scaling numerik
scaler = StandardScaler()
features_scaled = scaler.fit_transform(features)

# Ubah kembali ke DataFrame
features_scaled_df = pd.DataFrame(features_scaled, columns=features.columns)

# Gabungkan kembali dengan target
df_scaled = pd.concat([features_scaled_df, target.reset_index(drop=True)], axis=1)

# Cek hasil akhir
df_scaled.head()
```

🧪 Phase 3: Feature Selection

- Three models were used to identify the most influential features:
    - Linear Regression Coefficients
    - Random Forest Feature Importances
    - XGBoost Feature Importances
- Consistently top-ranked features across all models:

```
Resource Allocation Efficiency     1.000000
Project Duration (days)            0.042610
Material Quantities                0.037555
Equipment Usage                    0.025672
Best Cost (BC)                     0.008783
```

    - Equipment Usage
    - Project Duration (days)
    - Material Quantities
    - Best Cost (BC)

🔄 Phase 4: Model Training

- Tested multiple models:
  - **Linear Regression**
  - **Random Forest Regressor**
  - **XGBoost Regressor**
- Evaluation metrics used: Root Mean Square Error (RMSE) and R^2 Score

---

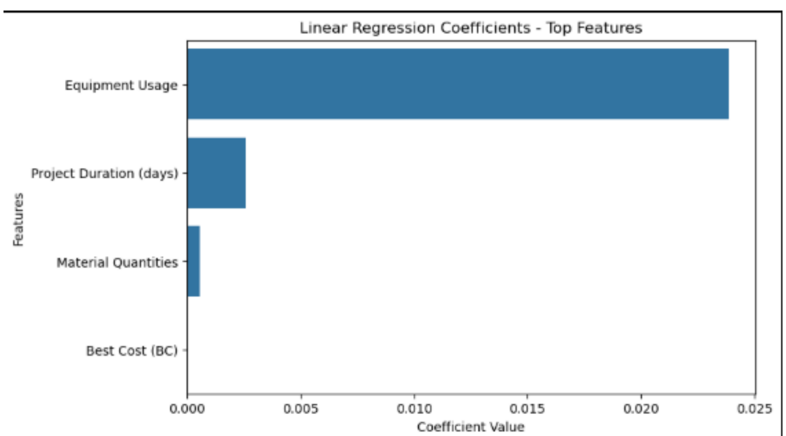
📊 Phase 5: Hyperparameter Tuning & Evaluation

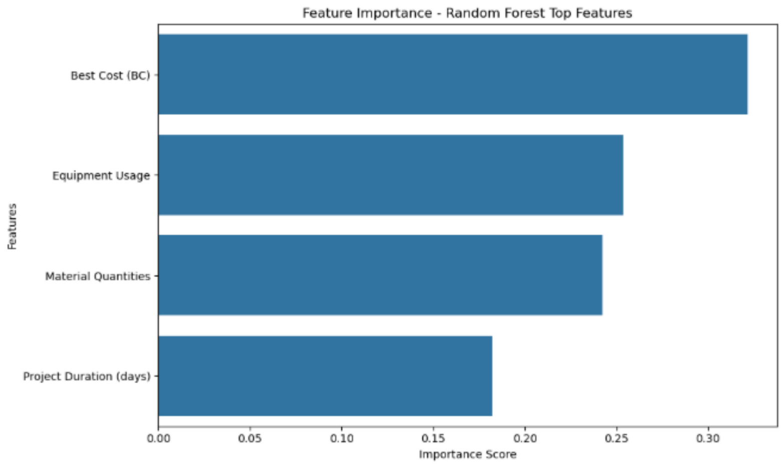
- Performed RandomizedSearchCV for XGBoost tuning.
- Visualized feature importance to improve model interpretability.
- Even after tuning, complex models did not outperform Linear Regression.

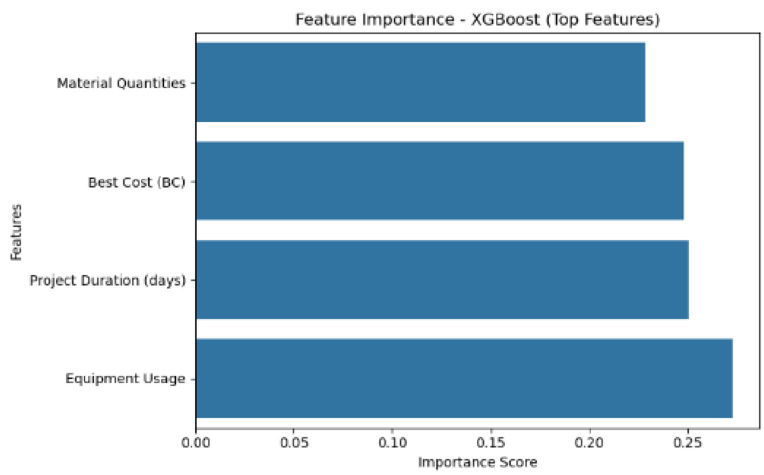
Model	**RMSE**	**R^2 Score**
Linear Regression	✅ **11.6087**	✅ **-0.0078**
Random Forest (Top Features)	**12.0634**	**-0.0883**
XGBoost (Top Features)	**13.2586**	**-0.3146**
XGBoost (Tuned)	**13.2012**	**-0.3033**

🔍 **Insight**: Despite all models yielding negative R^2 scores, Linear Regression achieved the best RMSE, indicating the relationship between input features and the target variable is relatively weak but more linear.

Comparison of Top Features Across Models (Linear Regression, Random Forest, XGBoost)







🔍 Interpretation:

The bar charts above illustrate the top feature importances identified by three different models: Random Forest, Linear Regression, and XGBoost.

- **Linear Regression** places the highest weight on **Equipment Usage**, followed by **Project Duration (days)**, with significantly lower weights for **Material Quantities** and **Best Cost (BC)**. This suggests that Equipment Usage has a strong linear correlation with Resource Allocation Efficiency.
- In contrast, both **Random Forest** and **XGBoost** consider **Material Quantities**, **Best Cost (BC)**, and **Project Duration (days)** as the most important features. These models show a more balanced contribution across these top features, indicating the presence of non-linear interactions.
- The slight differences in importance ranking highlight that **Linear Regression** focuses on linear relationships, whereas **Random Forest** and **XGBoost** can capture more complex, non-linear patterns.

📈 Conclusion

While Linear Regression yielded the best performance metrics (lowest RMSE, highest R^2), Random Forest and XGBoost provided richer interpretations of feature importance through non-linear relationships. This emphasizes the trade-off between model simplicity and interpretability versus flexibility and complexity.

More complex models like Random Forest and XGBoost did not outperform the simpler Linear Regression model. This suggests:

- The data has weak predictive power for non-linear models.
- Adding more high-impact features (e.g., managerial, operational, or BIM-driven variables) could improve prediction performance in future iterations.