MACHINE LEARNING APPROACHES FOR PRODUCTIVITY ESTIMATION OF BULLDOZERS

CASE STUDY PHASE 2

TUGBERK ERDOGMUS
SINEM KARAOMEROGLU
SARA MORADI JAFARBEGLOO
UGUR EYLIK

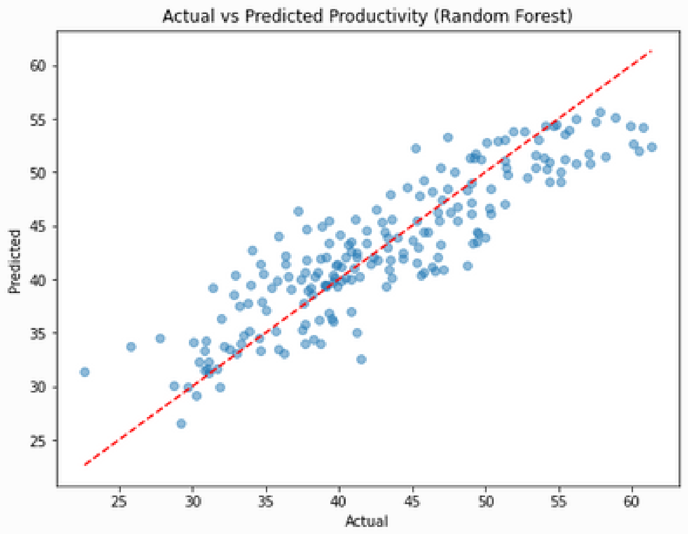
MACHINE LEARNING APPROACHES FOR PRODUCTIVITY ESTIMATION OF BULLDOZERS

01

What We Did Before?

Uniform Distribution & Linear Regression

Linear Regression R^2: 0.8291
Linear Regression MSE: 11.2443

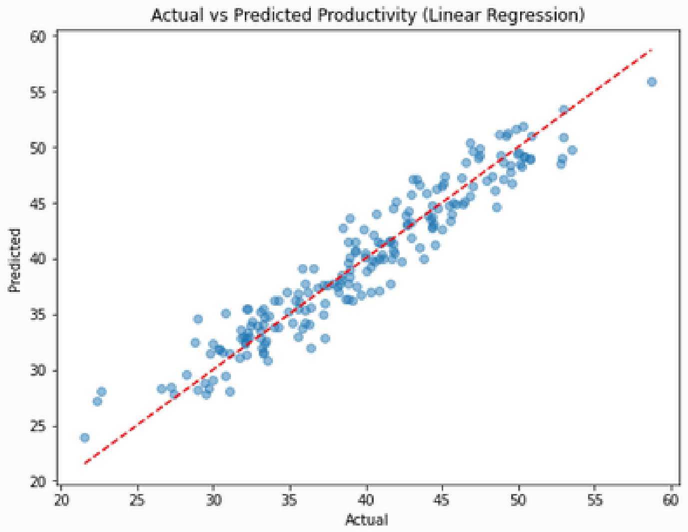


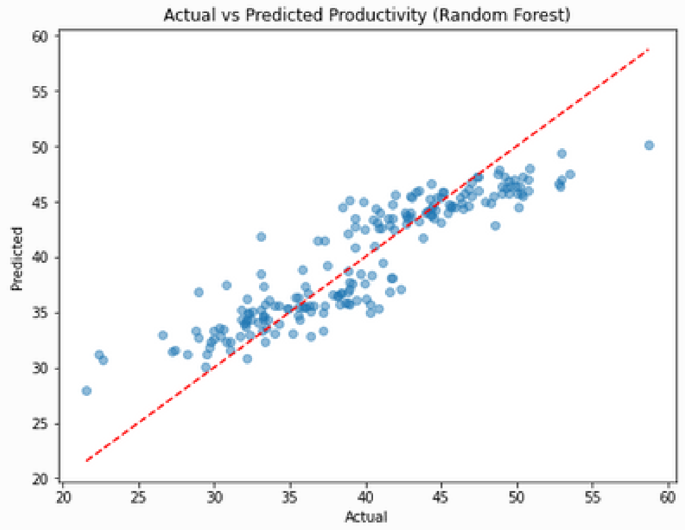
Uniform Distribution & Random Forests

Random Forest R^2: 0.7840
Random Forest MSE: 14.2111

Monte Carlo Simulation & Linear Regression

Linear Regression R^2: 0.9160
Linear Regression MSE: **4.1119**



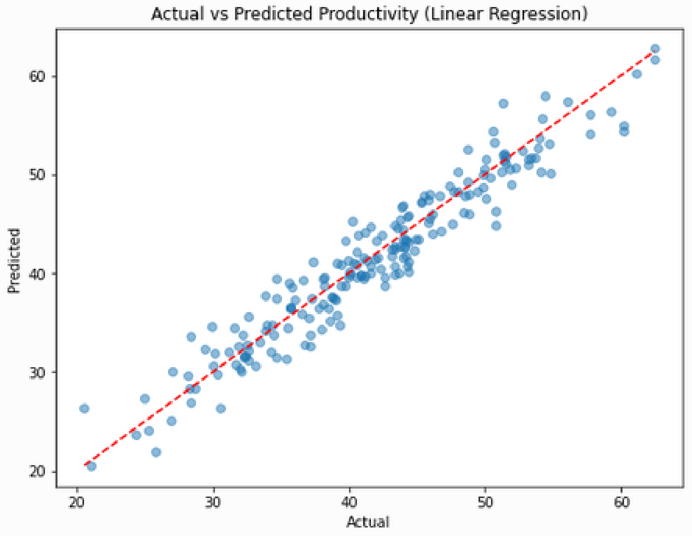


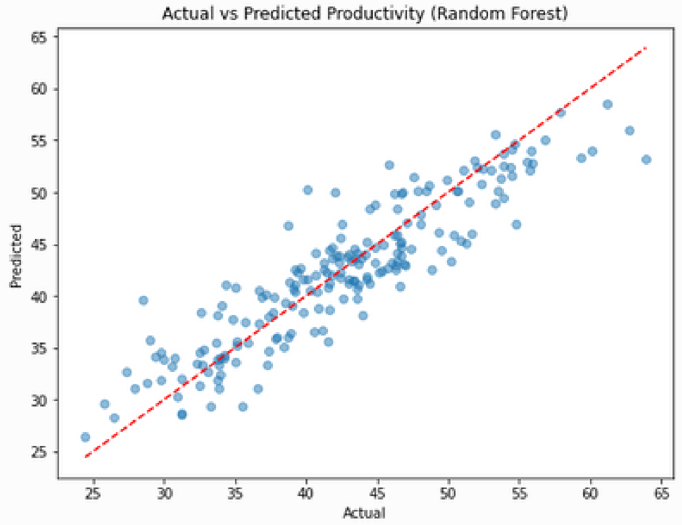
Monte Carlo Simulation & Random Forests

Random Forest R^2: 0.8055
Random Forest MSE: 9.5275

Latin Hypercube Sampling & Linear Regression

Linear Regression R^2: **0.9234**
Linear Regression MSE: 5.3756





Latin Hypercube Sampling & Random Forests

Random Forest R^2: 0.8195
Random Forest MSE: 11.2527

Concepts will be focused on

02

- Feature Engineering
- Random Forest
- Feature importance in Random Forest
- Regularization
- Ridge Regression
- Lasso Regression
- Elastic-Net Regression

Feature Engineering

What is Feature Engineering?

Feature engineering is the process of using domain knowledge to create or transform features (variables) that help machine learning models perform better. It involves generating new features, modifying existing ones, and selecting the most relevant features to improve the accuracy and effectiveness of predictive models. Feature engineering can significantly enhance the performance of regression models and other machine learning algorithms.

Why We Use the Feature Engineering?

- **Improves Model Performance:** Properly engineered features can improve the accuracy, precision, and overall performance of a regression model by providing it with more relevant and discriminative information.
- **Reduces Overfitting:** By selecting the most relevant features and removing noise, feature engineering helps in reducing overfitting, making the model more generalizable to unseen data.
- **Simplifies Models:** Effective feature engineering can lead to simpler models that are easier to interpret and understand, without compromising on performance.
- **Handles Non-Linearity:** Regression models, especially linear regression, assume a linear relationship between features and the target variable. Feature engineering can help to model non-linear relationships by transforming features appropriately.
- **Improves Convergence:** For algorithms that rely on iterative optimization (like gradient descent), having well-scaled and normalized features can lead to faster and more stable convergence.

Random Forest

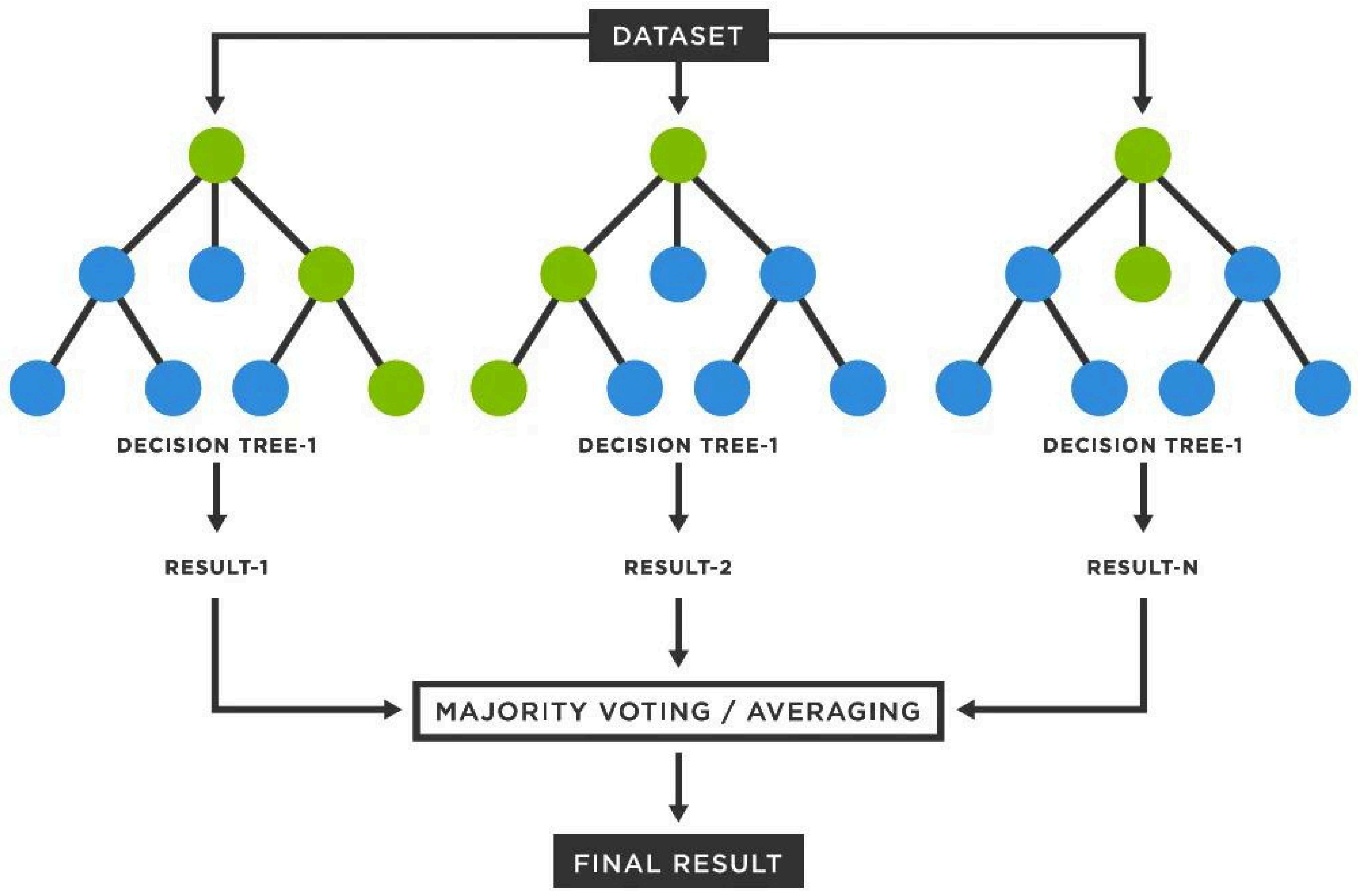
02

Used for classification and regression tasks

Builds multiple decision trees and merges their outcomes

Random Forest is used because:

1. High Performance and Reliable Accuracy
2. Reduction of Overfitting
3. Stability and Robustness
4. Handling High Dimensional Data
5. Feature Importance Estimation
6. Handling Missing Data



Feature Importance In Random Forest

02

Used to measure the significance of each feature

Helps to understand which features contribute the most to the model's predictions.

There are two common ways to calculate feature importance in a random forest:

- Mean Decrease in Impurity (MDI)
- Permutation Feature Importance

By analyzing feature importance, you can:

- Focus on the most relevant features
- Identify irrelevant features
- Gain insights into your data

	importance
Dozing distance (m) Ripper used?_1	0.137197
Average temperature during operation (°C) Rippe...	0.082818
Total service life time (hours) Ripper used?_1	0.063476
Number of consecutive operational days Ripper u...	0.053587
Ground grade (%) Ripper used?_1	0.044007
Dozing distance (m) Operation time_3	0.037214
Dozing distance (m) Average temperature during ...	0.029165
Dozing distance (m) Operation time_2	0.028696
Dozing distance (m) Type of blade_4	0.025764
Dozing distance (m) Type of blade_3	0.024649
Type of blade_4 Ripper used?_1	0.022516
Ripper used?_1 Operation time_3	0.020407
Dozing distance (m) Maximum blade capacity (m³)...	0.020332
Dozing distance (m) Type of blade_2	0.018802
Dozing distance (m) Maximum blade capacity (m³)...	0.018063
Total service life time (hours) Dozing distance...	0.017663
Dozing distance (m)^2	0.017225
Number of consecutive operational days Dozing d...	0.016895
Dozing distance (m) Maximum blade capacity (m³)...	0.015735
Ripper used?_1 Operation time_2	0.015505

Regularization

02

- While training models **->** **Problems: Underfitting and Overfitting.**
- Underfitting happens when the model neither performs well on the training data nor testing data.
- Overfitting happens when the model performs
  - well on the training data
  - not well on testing data.

- Regularization used to reduce errors caused by overfitting of the data

$$Y = X_0 + B_1X_1 + B_2X_2 + \cdots + B_nX_n$$

- **Least Squares** determines values for the parameters in the equation and minimizes Residual sum of squares (RSS)
- **Residual sum of squares (RSS)** measures how well a linear regression model matches training data. It is represented by the formulation:

$$RSS = \sum_{i=1}^{n} (Y_i - \hat{Y}_l)^2$$

- Coefficients for each independent variable are calculated
- Best-fitting line is created for a given dataset

Ridge Regression

Residual Sum of Squares + λ * (Sum of the squared value of the magnitude of coefficients)

$$L2 = ||B||^2 = B_1^{\,2} + B_2^{\,2} + \cdots + B_n^{\,2}$$

$$RSS_{L2} = \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 + \lambda \sum_{j=1}^{P} B_j^2$$

- L2 regularization technique.
- **To prevent multicollinearity in regression analysis .**
- **To reduce Model Complexity.**

02

- LS -> High coefficients -> the model's output becomes sensitive to minor changes in input data.
- The **model** has **overfitted on a specific training set.**

- Ridge regression shrinks coefficients by introducing a **penalty term** into the RSS function.
- Lambda is the penalty term that denotes the amount of shrinkage (zero to +infinity)

- Ridge estimator calculates new regression coefficients that reduce RSS.
- Predictor's effects are minimized and overfitting is reduced.

- Ridge regression does not shrink every coefficient by the same value.
- As λ increases, high-value coefficients shrink at a greater rate than low-value coefficients.
- **High-value coefficients are thus penalized greater than low-value coefficients.**

****Coefficients are shrunk towards zero but not exactly to zero.

Residual Sum of Squares + λ * (Sum of the absolute value of the magnitude of coefficients)

Lasso Regression

02

- Lasso (**L**east **A**bsolute **S**hrinkage **S**elector **O**perator)
- L1 regularization technique.
- To prevent multicollinearity in regression analysis .
- To reduce Model Complexity.

- Lasso regression shrinks coefficients by introducing a **penalty term** into the RSS function.
- Lambda is the penalty term that denotes the amount of shrinkage (zero to +infinity)

- The larger the value, the more aggressive the penalization is.
- Some coefficients might become zero and get eliminated from the model.
- **Feature selection** using a Shrinkage method (Penalized regression method).

- Used when number of features are more, because it automatically does feature selection.

RESULT:
Less important features in a dataset are eliminated by penalty.
More interpretable, simpler model by focusing on significant predictors.

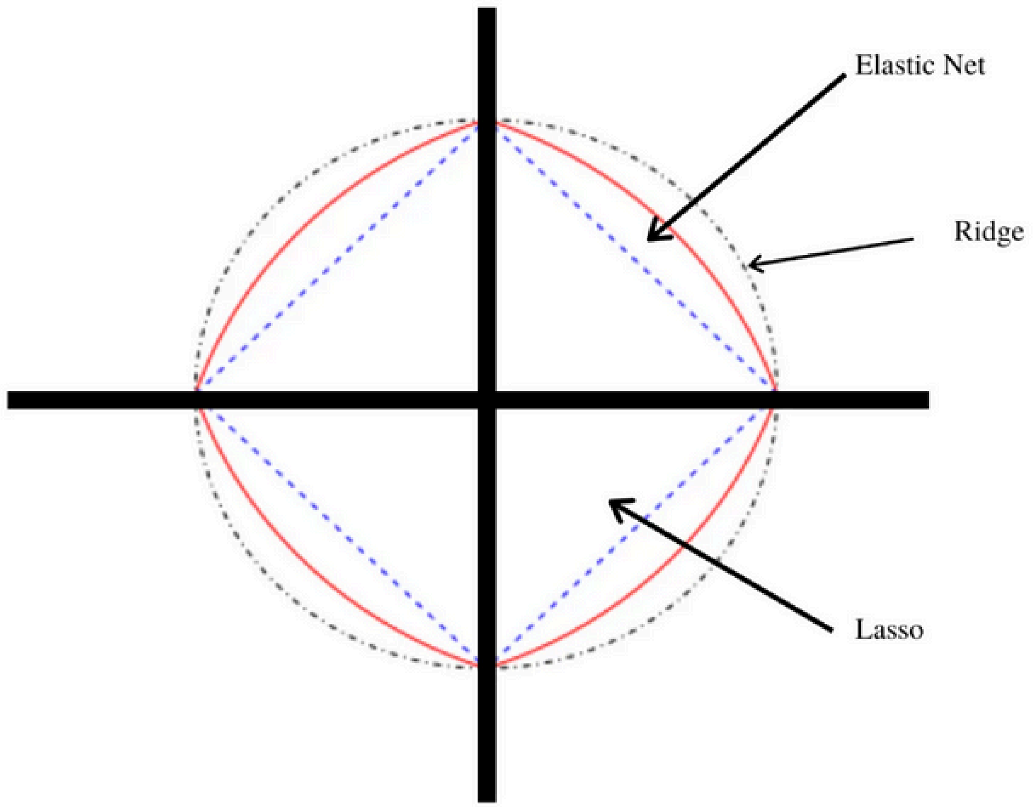
****Coefficients may be shrunk exactly to zero and get eliminated.

Elastic-Net Regression

- Elastic net is a combination of both L1 and L2 regularization.
- First finds the ridge regression coefficients.
- Then conducts the second step by using a lasso shrinkage of the coefficients.
- The elastic net method improves lasso’s limitations.
- If the variables are highly correlated groups, lasso tends to choose one variable from such groups and ignore the rest entirely.

- Elastic-Net Regression groups and shrinks the parameters associated with the correlated variables and leaves them in equation or removes them at all once.

02



$$\text{in}\left(||Y - X\theta||_2^2 + \lambda_1||\theta||_1 + \lambda_2||\theta\right.$$

Methodology

03

1- Define Sample Values and Feature Ranges

2- Dataset Generation

- Uniform Distribution(Noise added)
- Monte Carlo Simulation(Noise added)
- Latin Hypercube Sampling(Noise added)

3- Preprocessing

- Polynomial Features

Calculation of Polynomial Features

X1	X2	$X1^2$	$X2^2$	$X1 \times X2$
1000	4.8	1000000	23.04	4800
2000	6.8	4000000	46.24	13600
3000	8.8	9000000	77.44	26400

$$y = \beta_0 + \beta_1 \cdot X1 + \beta_2 \cdot X2 + \epsilon$$

$$y = \beta_0 + \beta_1 \cdot X1 + \beta_2 \cdot X2 + \beta_3 \cdot X1^2 + \beta_4 \cdot X2^2 + \beta_5 \cdot (X1 \times X2) + \epsilon$$

Advantages

1. Capturing Non-Linearity
2. Flexibility

4- Feature Selection

- Variance Inflation Factor (VIF): Measures how much the variance of a regression coefficient is inflated due to multicollinearity among the features.

Calculation of VIF

1. Regression of Each Feature
2. Coefficient of Determination
3. Compute VIF

$$VIF_i = \frac{1}{1-R^2}$$

VIF = 1: No correlation between the feature and others.

VIF > 1: Some correlation, but not severe.

VIF > 5-10: High multicollinearity, indicating that the feature might be redundant.

	feature	VIF
0	const	5.086709e+01
1	Total service life time (hours)	1.076551e+01
2	Number of consecutive operational days	1.079333e+01
3	Ground grade (%)	1.051463e+01
4	Dozing distance (m)	1.054788e+01
..	...	...
115	Ripper used?_1 Operation time_2	3.375446e+00
116	Ripper used?_1 Operation time_3	3.490631e+00
117	Operation time_2^2	1.411074e+07
118	Operation time_2 Operation time_3	NaN
119	Operation time_3^2	3.265391e+09

Methodology

03

Model-Specific Steps

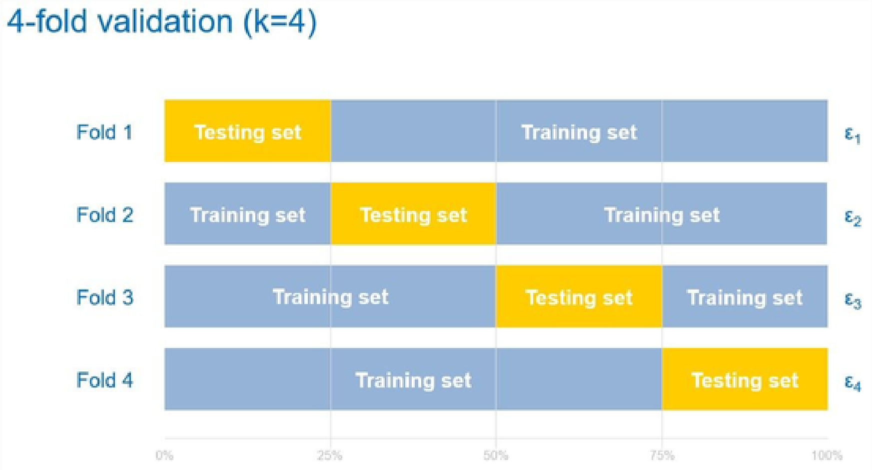
5- Train-Test Split

6- Model Implementation

1. **Multiple Linear Regression**
2. **Ridge Regression**
3. **Lasso Regression**
4. **Elastic Net**
5. **Random Forest**

Each Model has the following;

- Hyperparameter Tuning with cross-validation(CV)




```
final_cv_scores = cross_val_score(final_lin_reg, X_train, y_train, cv=5, scoring='r2')
```

```
Linear Regression CV R²: 0.6793 ± 0.0052
```

- Fit an Ordinary Least Squares(OLS) model to inspect p-values and iteratively remove features with high p-values

The method minimizes the sum of the squared differences between the observed dependent variable values and those predicted by the linear function.

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

```
Df Model:                                                    89
Df Model:                                                    47
```

```
                                                                  coef    std err          t      P>|t|      [0.025
0.975]
-----------------------------------------------------------------------------------------------------------------------
-----------
const                                                          38.8711      0.216    180.101      0.000      38.448
39.294
Total service life time (hours)^2                               0.0429      0.080      0.537      0.592      -0.114
0.199
Total service life time (hours) Number of consecutive operational days   -0.0268      0.071     -0.376      0.707      -0.167
0.113
```

- Retaining with the best parameters and evaluation of the final model

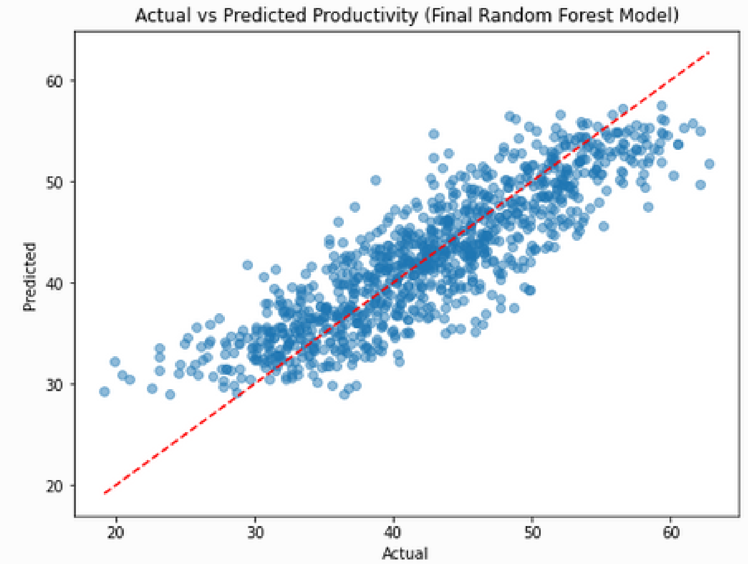
Results

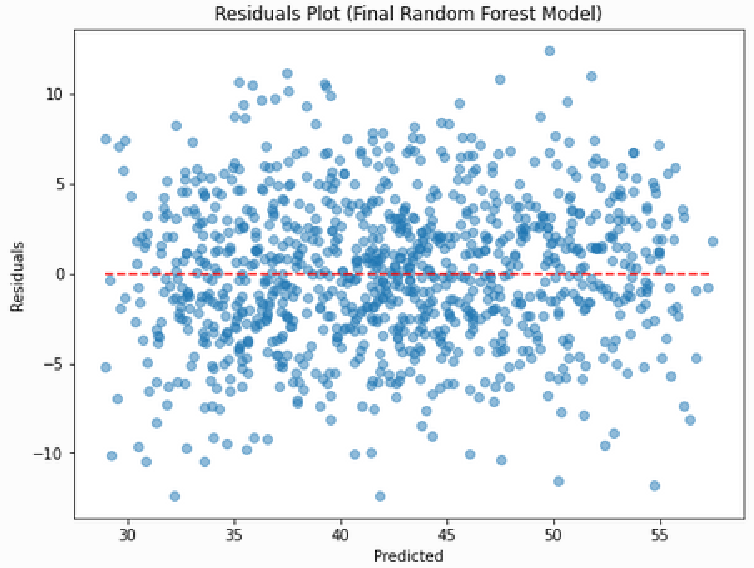
Number of data(n)= 5000

03

Uniform Distribution Random Forest

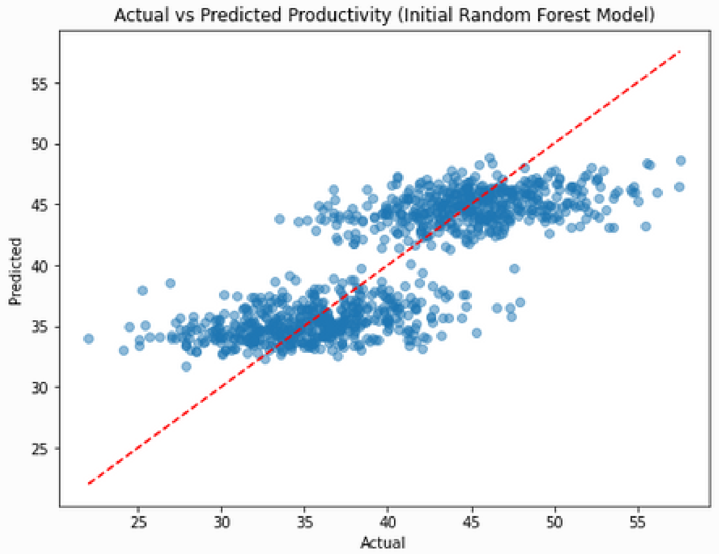
```
Final Random Forest R²: 0.7486
Final Random Forest MSE: 16.6451
Final Random Forest Adjusted R²: 0.7427
```

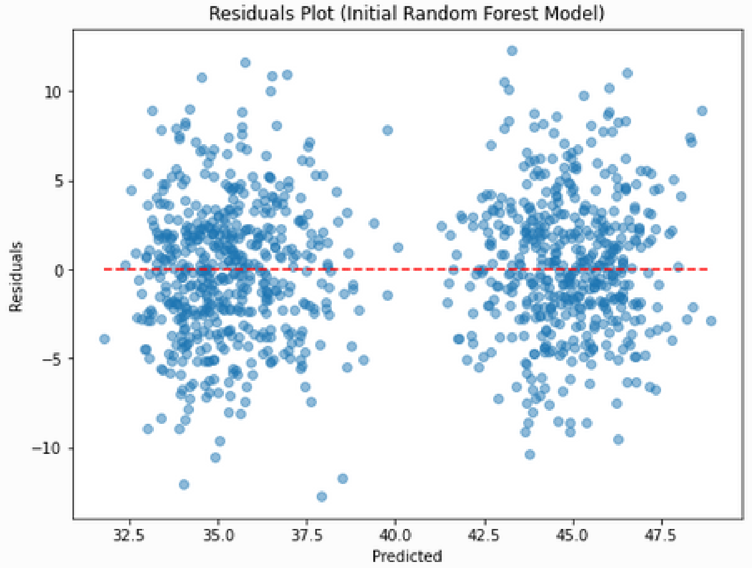




Monte Carlo Simulation Random Forest

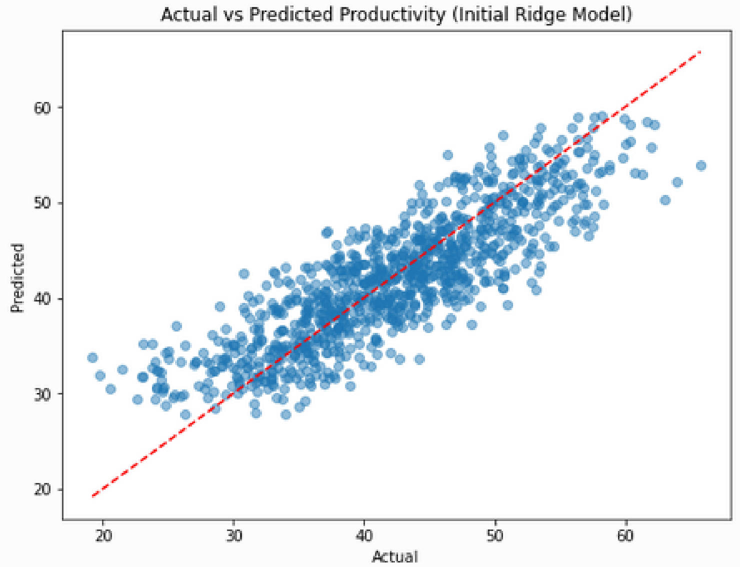
```
Final Random Forest R²: 0.6145
Final Random Forest MSE: 16.3702
Final Random Forest Adjusted R²: 0.6094
```

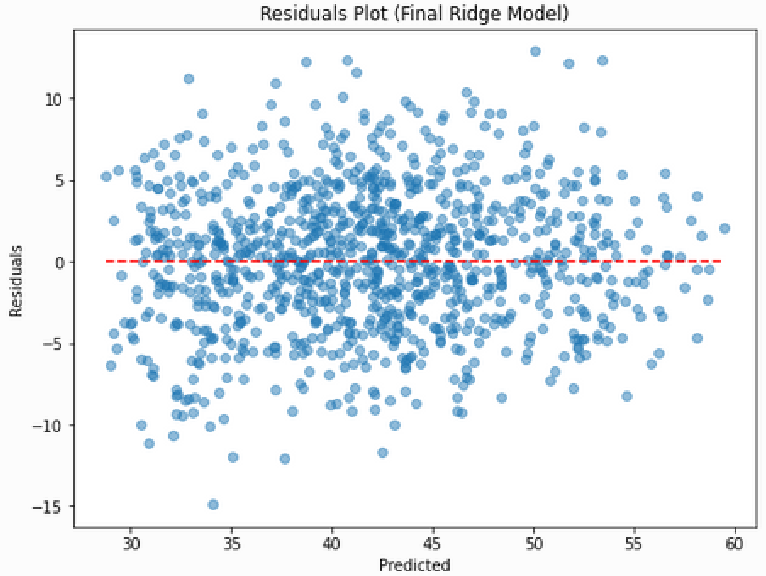




Latin Hypercube Sampling Ridge Regression

```
Final Ridge Regression R²: 0.7205
Final Ridge Regression MSE: 18.5414
Final Ridge Regression CV R²: 0.7084 ± 0.0143
```





Conclusion

03

Number of data = 1000

- Better results with Linear Regression when the dataset is **less complex** and **easier to implement** and **interpret**
- The combination of **Latin Hypercube Sampling** and **Linear Regression** gave us the best result
- Random Forest could not get better results compared to Linear regression

Number of data = 5000

- Random Forest is **more flexible** and **better at modeling complex patterns** in the data
- The **Random Forest with Uniform Distribution** yielded the **best overall performance**
- As the amount of our **data increases**, **Monte Carlo Simulation** creates a more **dispersed dataset** and makes it more **difficult to build a model** on it
- The **Ridge Regression with Latin Hypercube Sampling** provided a **strong balance of performance and robustness**

What will we do in the final?

- Multicollinearity problem

```
[2] The smallest eigenvalue is 3.99e-29. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

- Focus on parameters

Thank You