Ben-Gurion University of the Negev

Is it possible to predict the processing time of manufacturing machines in the field of semiconductor processing?

February, 2021

Lecturer: Prof. Yisrael Parmet

MSc Students: Gavan Goldberg, Tomer Panker, Tom Landman

ABSTRACT

The semiconductor industry is the aggregate collection of companies engaged in the design and fabrication of semiconductors. It formed around 1960, once the fabrication of semiconductor devices became a viable business. Semiconductors are materials that have a conductivity between conductors (e.g., metallic copper) and nonconductors or insulators (e.g., glass). Due to their role in the fabrication of electronic devices, semiconductors play an essential part in our lives. Therefore, semiconductor manufacturing is characterized by manufacturing processes that require strict process control and a high cost of machinery. Predicting the various machines' processing time allows rapid identification of changes in the production process and optimal utilization of existing machines. This study aimed to examine the factors that may affect processing time and train a predictive model on the relevant features. This study is based on production data as documented in the machines' log files. During the study, we conduct experiments using the 10-Fold-Cross-Validation (CV) procedure to examine both parametric-based and non-parametric-based algorithms such as Linear Regression (LR) [1], Support Vector Regressor (SVR) [2] [3], Random Forest (RF) [4], and XGBoost [5]. After examining the different models, we compared their results according to several regression metrics and found that the prediction model that outperformed the rest was XGBoost with an MSE of 0.003.

1. INTRODUCTION

Semiconductor materials have many uses in electronics. These materials are used extensively in the computer processor manufacturing industry in general and in Intel in particular. The global semiconductor manufacturing market is estimated at $ 489,820,000,000, and therefore, there is great financial significance to every percentage improvement made in this field. A semiconductor is a material whose electrical conduction properties are in the wide range between those of conductors and resistors. The conduction properties vary greatly depending on external factors such as temperature and exposure to light. The unique production process has several main characteristics: 1) High cost of machines; 2) Each machine is required to perform many operations; 3) High variability in the production process. As shown in **Fig. 1** the most common semiconductors are Resistors, Capacitor, Diodes, Transistors, Integrated Circuits (ICs), and Operational Amplifiers (op-amp).

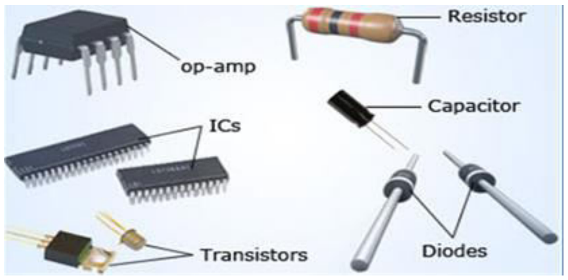


Fig. 1. Types of semiconductor devices.

Business-wise, there is tremendous global competition in the semiconductor manufacturing industry. As a result, a company that fails to manage the production process efficiently will not utilize the machines optimally, leading to significant financial losses. A necessary condition for efficient utilization of the production machines is discovering how machine-related factors affect the processing time and can provide an accurate prediction capability of the expected processing time.

2. RESEARCH GOALS

This study aims to predict manufacturing machines' processing time in the semiconductor industry and understand the factors influencing the processing time. This study is intended to enable optimal utilization of the production machines thanks to a more accurate prediction of the processing time on the one hand and the identification of the factors that prolong the processing time on the other hand.

Our research question is to examine if it is possible to predict manufacturing machines' processing time in semiconductor processing based on the machine's logs?

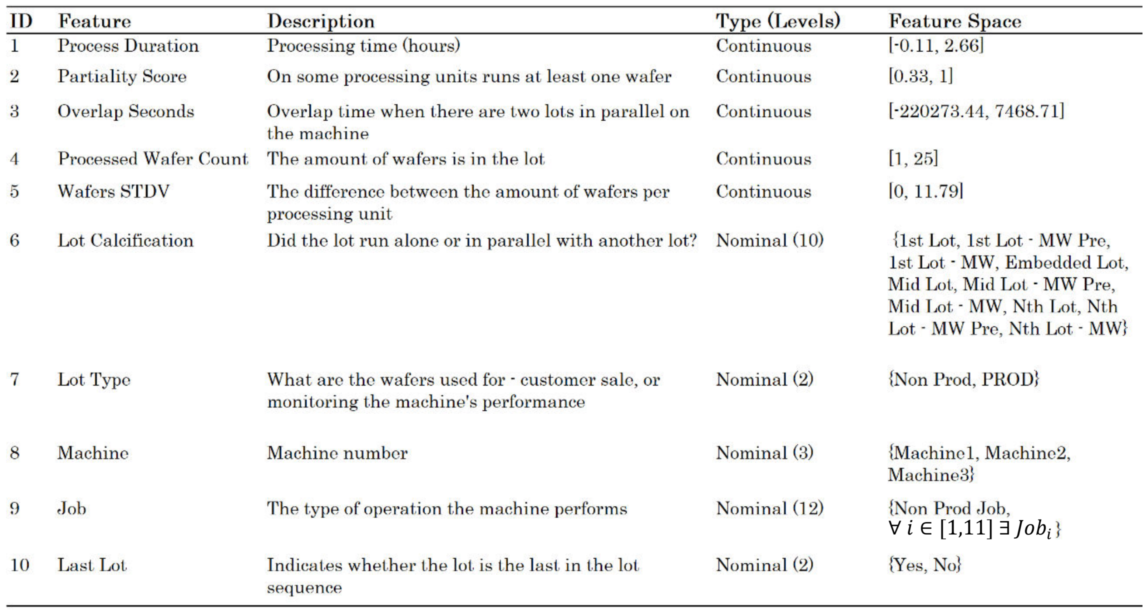
3. METHODS

3.1. Data Collection

The research data we used were taken from the log files of the production machines. Production data is stored in these logs in a structured manner as part of an orderly Extract Transform Load (ETL) process [6]. From a data quality perspective, the log files are characterized with a minor chance of missing values, and indeed all 9086 records we collected were not contained any missing data.

Table 1 below summarizing the features on which we relied in this study:

Table 1
Details and description of the existing features in the data.



ID	Feature	Description	Type (Levels)	Feature Space
1	Process Duration	Processing time (hours)	Continuous	[-0.11, 2.66]
2	Partiality Score	On some processing units runs at least one wafer	Continuous	[0.33, 1]
3	Overlap Seconds	Overlap time when there are two lots in parallel on the machine	Continuous	[-220273.44, 7468.71]
4	Processed Wafer Count	The amount of wafers is in the lot	Continuous	[1, 25]
5	Wafers STDV	The difference between the amount of wafers per processing unit	Continuous	[0, 11.79]
6	Lot Calcification	Did the lot run alone or in parallel with another lot?	Nominal (10)	{1st Lot, 1st Lot - MW Pre, 1st Lot - MW, Embedded Lot, Mid Lot, Mid Lot - MW Pre, Mid Lot - MW, Nth Lot, Nth Lot - MW Pre, Nth Lot - MW}
7	Lot Type	What are the wafers used for - customer sale, or monitoring the machine's performance	Nominal (2)	{Non Prod, PROD}
8	Machine	Machine number	Nominal (3)	{Machine1, Machine2, Machine3}
9	Job	The type of operation the machine performs	Nominal (12)	{Non Prod Job, $\forall\, i \in [1,11]\ \exists\, Job_i$}
10	Last Lot	Indicates whether the lot is the last in the lot sequence	Nominal (2)	{Yes, No}

3.2. Exploratory Data Analysis

3.2.1 Correlation Matrix

The correlation matrix is a table showing correlation coefficients between variables. Each cell in the table represents the correlation strength between two variables. The correlation matrix in **Fig. 2** below describes all the continuous variables in our study. We can identify two relatively strong correlations between the IVs (independent variables) and the DV (dependent variable) from the matrix above. We can see a strong positive correlation between Processed Wafer Count (IV) and Process duration (DV) driven by the increasing amount of wafers machine process. Additional significant correlation can be found between Wafers STDV (IV) and Process duration (DV). From a domain perspective, this relationship is driven by inefficient wafer split through the machine that triggers other technical limitations.

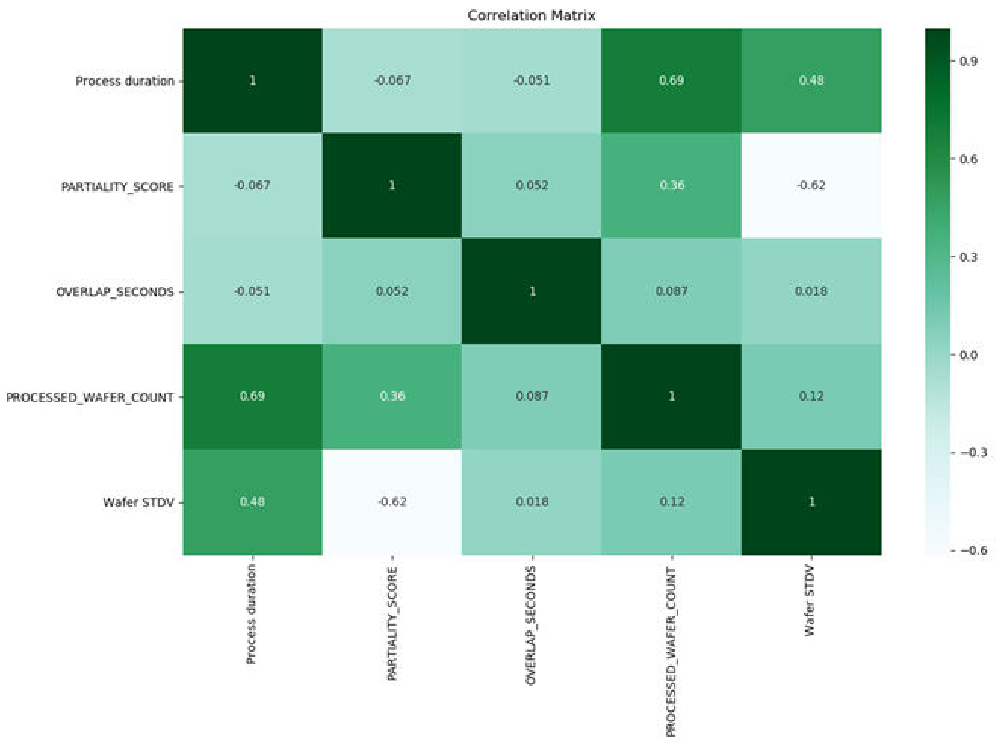


Fig. 2. Pearson correlation matrix of continuous features.

3.2.2 Histograms & Scatter Plots

In **Fig. 3** below, we display bivariate relations between the continuous features whereas, histograms are presented on the diagonal.

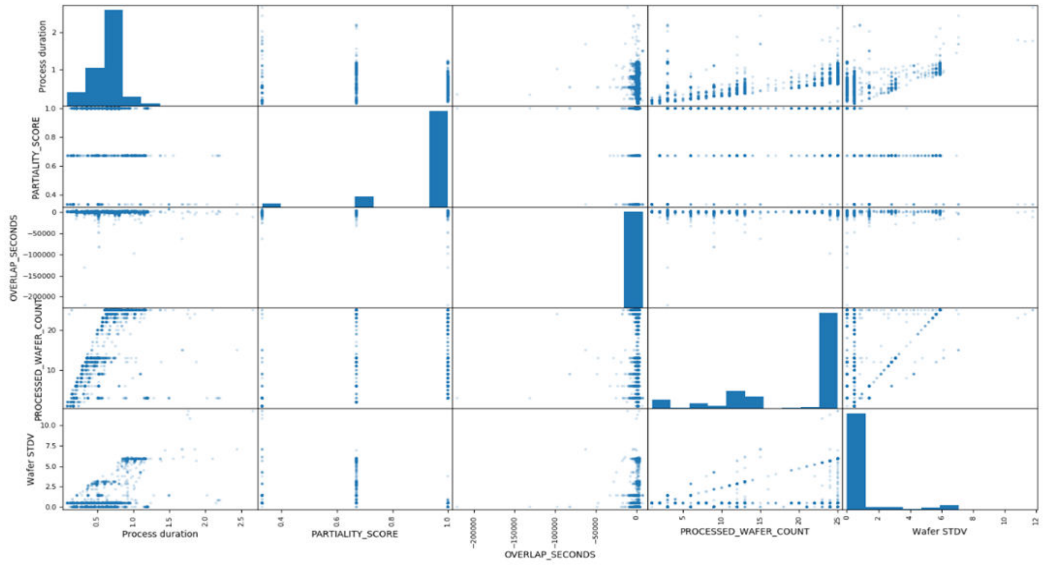


Fig. 3. Matrix of Histograms (on the diagonal) and Scatter plots (off the diagonal) of continuous features.

The scatter plot in **Fig. 4** below shows the correlation between Overlap and Process duration. When focusing on the dark green dots (Partiality score =1), we can identify an interesting relationship between the overlap and the process duration changes in the Overlap variable's different ranges. Below zero, there is no correlation between the variables. However, in the range of 0 to 350 seconds, there is a high negative correlation. From a domain perspective, this is caused by adding additional constraints on top of the machine bottleneck. Above 350 seconds, the machine processing unit became the bottleneck, and the correlation change to zero.

Also, note that overlap is the interval time in which two lots are placed on the machine in parallel. In contrast, negative overlap reflects the interval time between lot arrival (i.e., no overlap).

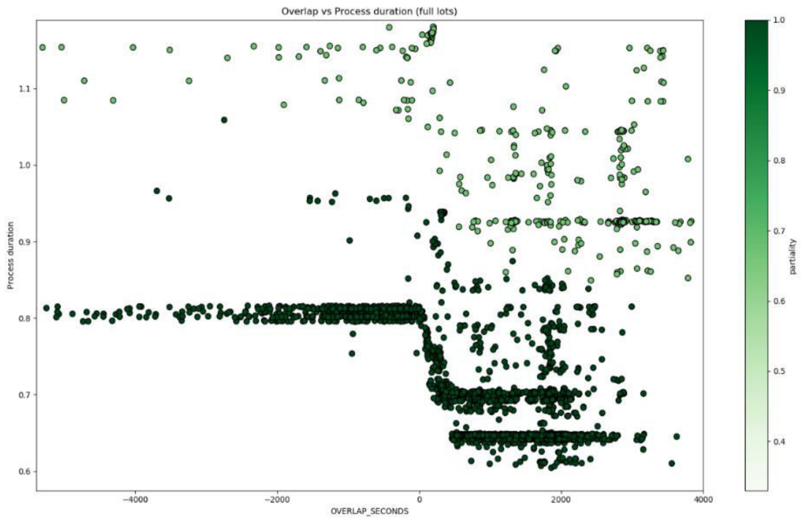


Fig. 4. Scatter plot of process duration as function of overlap (seconds).

3.2.3 Box Plot

Fig. 5 below describes the process duration for each machine. We can see that the median and the IQR is similar across all machines. However, the first machine distribution (i.e., Machine 1) is more skewed to the right than other machines, resulting from technical issues on the machine.

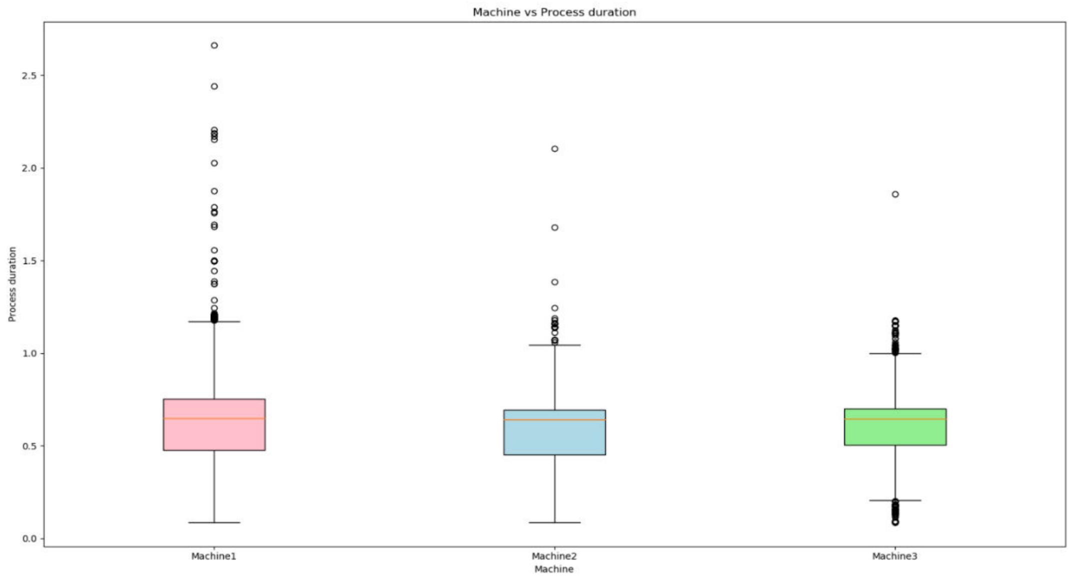


Fig. 5. Box plot of process duration (hours) distributed by 3 production machines.

3.3. Design of Experiments

In order to answer our research question, we examined various models whose purpose was to predict the batch's processing time given the other features. Since the batch processing time is a continuous variable, we are dealing with a regression problem. Therefore, we selected several models that know how to handle regression problems and compared their performance using the K Cross-validation procedure to minimize the noise in the results by averaging ten different folds results. Since each fold consists of records that do not exist in other folds, the results' average will make it possible to identify the most robust model with the best generalization capabilities relative to the other models. To evaluate the results, we used several regression-based metrics: Mean Absolute Error (MAE), Mean Squared Error (MSE), Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), and R^2.

3.4. Models

In this subsection, we present our chosen regression-based algorithms used to address the task of predicting the process duration time given the rest of the other collected production features, followed by their results. We divided this subsection into two types of models: Parametric and non-parametric [7]. A learning model that summarizes data with a finite set of parameters, θ, of fixed size (independent of the number of training examples) is called a parametric model. Hence, no matter how much data is thrown at a parametric model, it will not change the amount of needed to be estimated. However, nonparametric models are suitable when dealing with many data with no prior knowledge, or in cases when selecting the right features is not the primary consideration or a constraint [8]. The amount of information that θ can capture about the data $\mathcal{D}$ can grow as the amount of data grows. This makes nonparametric models more flexible.

3.4.1 <u>Parametric Models</u>

In this subsection, we present four parametric models that we tested in order to address our regression task of predicting the semiconductors manufacturing process's duration based on the rest of the features. We selected the following models: Linear Regression, Ridge Regression [9], Lasso Regression [10], and Support Vector Machine Regressor [2] [3].

3.4.1.1 *Linear Regression*

This subsection briefly describes the linear regression model used to predict the machines' processing duration. This model was constructed using the LinearRegression class from the *sklearn*[1] package written in Python. The definition of the model is according to **Eq. 1** below:

$$\boldsymbol{y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + \varepsilon_{i,}} \quad \forall \quad \boldsymbol{i = 1, \dots, n} \ (1)$$

Where y represents the processing time, X is the features matrix, β is the regression coefficients vector, and ε represents the random error. Also, note that the linear regression model assumptions are listed below (see Eq. 2-4 below):

I. **Linearity:** The relationship between X and the mean of Y is linear.

II. **Homoscedasticity:** The variance of residual is the same for any value of X (see **Eq. 2** below).

$$\forall\, i\; V(\varepsilon_i) = \sigma^2 \ (2)$$

[1] https://scikit-learn.org/stable

III. **Independence:** Observations are independent of each other (see **Eq. 3** below).

$$\forall\, i \neq j\; COV(\varepsilon_i, \varepsilon_j) = 0 \quad (3)$$

IV. **Normality:** For any fixed value of X, Y is normally distributed (see **Eq. 4** below).

$$\forall\, i\; E(\varepsilon_i) = 0 \quad (4)$$

The following is the hypothesis of the linear regression model:

$$H_0:\; \beta_j = 0$$

$$H_1:\; \beta_j \neq 0$$

During data preprocessing, we converted the nominal variables to dummies using the one-hot-encoding representation. That is, to appropriately apply the linear model to the desired data. The converted variables are *Lot Type*, *Last Lot*, *Lot Calcification*, *Job*, and *Machine*.

In the next subsection, we elaborate on the linear models' optimization process and our selected strategies.

3.4.1.2 *Linear, Lasso, Ridge, and Polynomial Regressions*

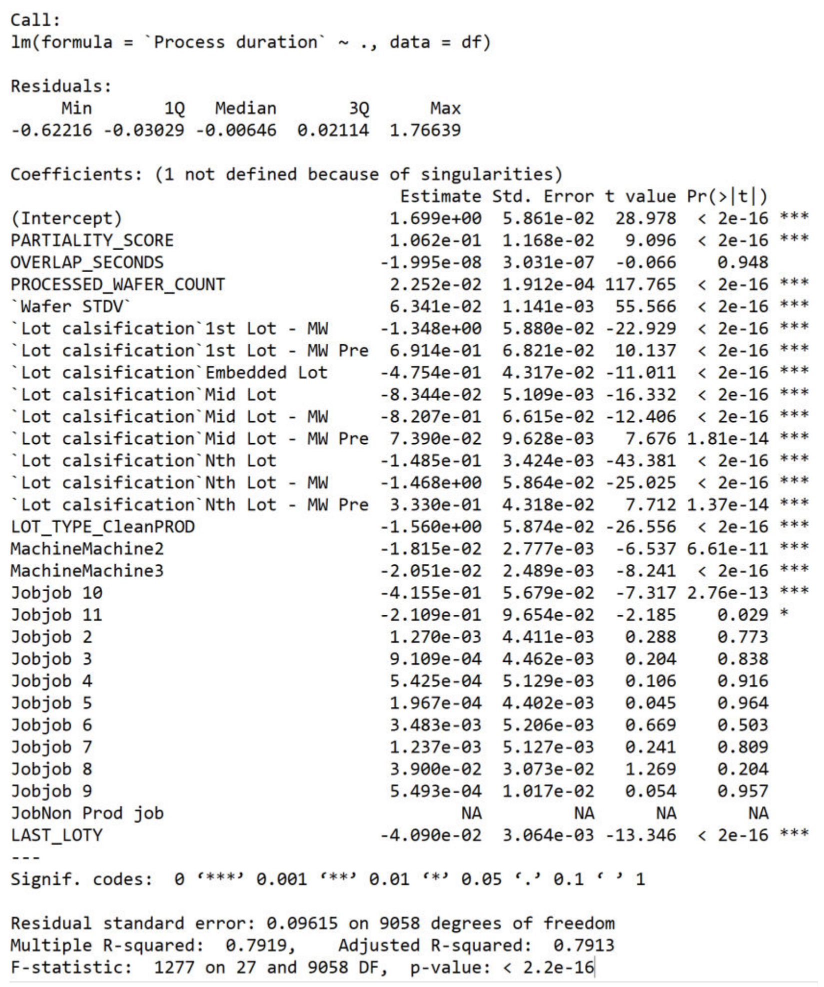
This subsection explains how we optimized the following linear models separately: Linear Regression, Ridge Regression (L_2 norm), and Lasso Regression (L_1 norm). All three models were optimized using a Random Search CV procedure. The hyperparameter to be tuned was the penalty λ that controls on the regularization strength. Regularization improves the conditioning of the problem and reduces the estimates' variance. Note that larger values imply a more robust regularization. We tested each one of the following 13 values for the penalty λ;

$$\lambda \in \{0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.5, 2.0\}.$$

After optimization, we ended up with three optimized models and compared their averaged metrics based on the 10-CV procedure without excluding any features. The linear regression model outperformed in most metrics and achieved the following scores: MAE score of 0.045, MSE of 0.01, R^2 of 0.821, AIC of 4452.84, and BIC of 4448.03. Second, the ridge regression model ($\lambda = 0.1$) achieved the following scores: MAE score of 0.045, MSE of 0.01, R^2 of 0.821, AIC of 4690, and BIC of 4685.01. Third, the lasso regression model ($\lambda = 0.01$) achieved the following scores: MAE score of 0.045, MSE of 0 .01, R^2 of 0.756, AIC of -4453, and BIC of -4448.02.

Moreover, since the linear regression model outperformed the rest models, we used it as a baseline for possible transformation and model selection techniques. As a result, as shown in **Fig. 6** below, we selected the LR model as our baseline for other parametric models and trained it on the entire data, and achieved the following scores: MAE score of 0.044, MSE of 0.009, R^2 of 0.792, AIC of -16740.185, and BIC of -16533.865.

Note that based on the LR model output, we can say that it is significant in terms of the Analysis Of Variance (ANOVA) test, which resulted in an F score of 1277 and achieved a $p_{value} < 2.2e-16$ with a significance level of $\alpha = 0.05$. Also, by viewing the resulted regression coefficients, it seems that most of the features achieved a significant t score with a significance level of $\alpha = 0.05$. However, in the case of *Overlap Seconds* and *Job,* it can be seen that their t scores are not significant.



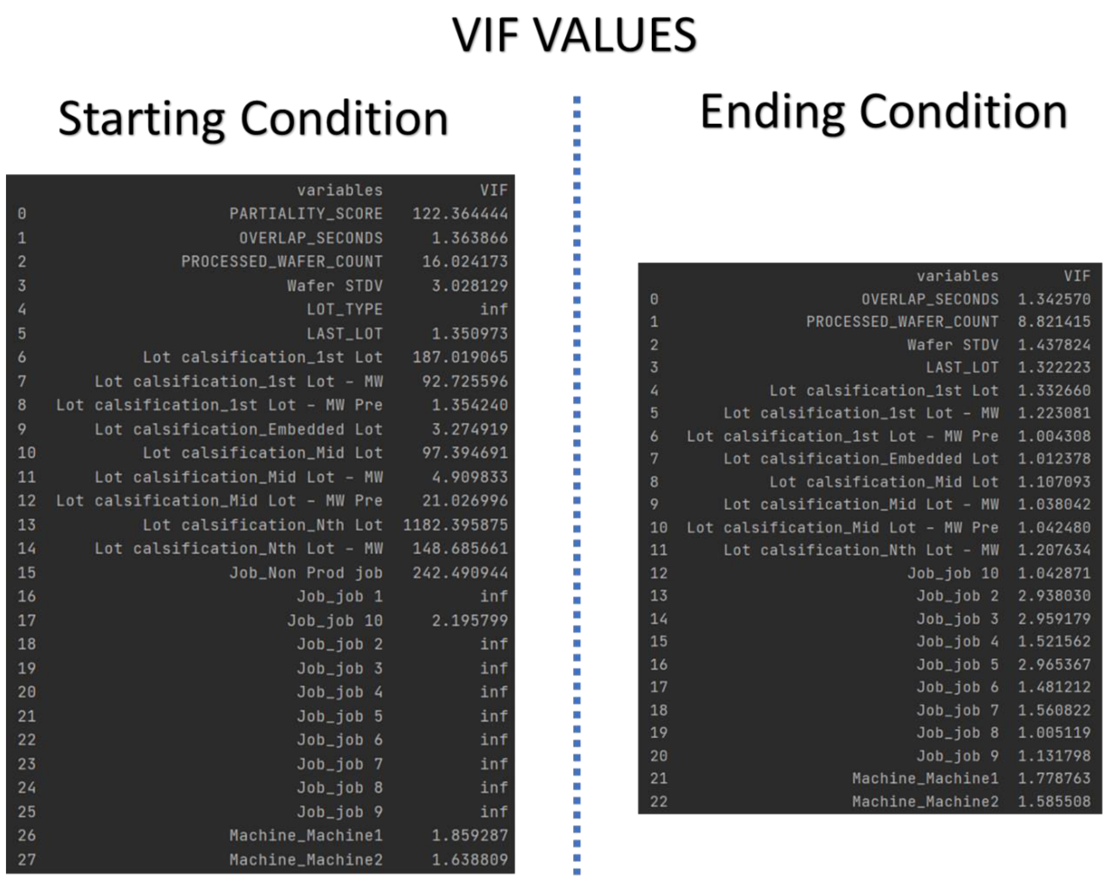
```
Call:
lm(formula = `Process duration` ~ ., data = df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.62216 -0.03029 -0.00646  0.02114  1.76639

Coefficients: (1 not defined because of singularities)
                                     Estimate Std. Error t value Pr(>|t|)
(Intercept)                         1.699e+00  5.861e-02  28.978  < 2e-16 ***
PARTIALITY_SCORE                    1.062e-01  1.168e-02   9.096  < 2e-16 ***
OVERLAP_SECONDS                    -1.995e-08  3.031e-07  -0.066    0.948
PROCESSED_WAFER_COUNT               2.252e-02  1.912e-04 117.765  < 2e-16 ***
`Wafer STDV`                        6.341e-02  1.141e-03  55.566  < 2e-16 ***
`Lot calsification`1st Lot - MW    -1.348e+00  5.880e-02 -22.929  < 2e-16 ***
`Lot calsification`1st Lot - MW Pre 6.914e-01  6.821e-02  10.137  < 2e-16 ***
`Lot calsification`Embedded Lot    -4.754e-01  4.317e-02 -11.011  < 2e-16 ***
`Lot calsification`Mid Lot         -8.344e-02  5.109e-03 -16.332  < 2e-16 ***
`Lot calsification`Mid Lot - MW    -8.207e-01  6.615e-02 -12.406  < 2e-16 ***
`Lot calsification`Mid Lot - MW Pre 7.390e-02  9.628e-03   7.676 1.81e-14 ***
`Lot calsification`Nth Lot         -1.485e-01  3.424e-03 -43.381  < 2e-16 ***
`Lot calsification`Nth Lot - MW    -1.468e+00  5.864e-02 -25.025  < 2e-16 ***
`Lot calsification`Nth Lot - MW Pre 3.330e-01  4.318e-02   7.712 1.37e-14 ***
LOT_TYPE_CleanPROD                 -1.560e+00  5.874e-02 -26.556  < 2e-16 ***
MachineMachine2                    -1.815e-02  2.777e-03  -6.537 6.61e-11 ***
MachineMachine3                    -2.051e-02  2.489e-03  -8.241  < 2e-16 ***
Jobjob 10                          -4.155e-01  5.679e-02  -7.317 2.76e-13 ***
Jobjob 11                          -2.109e-01  9.654e-02  -2.185    0.029 *
Jobjob 2                            1.270e-03  4.411e-03   0.288    0.773
Jobjob 3                            9.109e-04  4.462e-03   0.204    0.838
Jobjob 4                            5.425e-04  5.129e-03   0.106    0.916
Jobjob 5                            1.967e-04  4.402e-03   0.045    0.964
Jobjob 6                            3.483e-03  5.206e-03   0.669    0.503
Jobjob 7                            1.237e-03  5.127e-03   0.241    0.809
Jobjob 8                            3.900e-02  3.073e-02   1.269    0.204
Jobjob 9                            5.493e-04  1.017e-02   0.054    0.957
JobNon Prod job                            NA         NA      NA       NA
LAST_LOTY                          -4.090e-02  3.064e-03 -13.346  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.09615 on 9058 degrees of freedom
Multiple R-squared:  0.7919,    Adjusted R-squared:  0.7913
F-statistic:  1277 on 27 and 9058 DF,  p-value: < 2.2e-16
```

Fig. 6. Linear Regression model (i.e., baseline) output.

We thought about several strategies to improve our model, such as dealing with multicollinearity, using feature selection techniques, validating model assumptions, and applying transformations. First, we tried to handle multicollinearity by calculating the *variance inflation factor* (VIF) values (See Eq. 5) for all the features. As shown in **Fig. 7.** below, we got several features with a high level of VIF (i.e., $VIF(\hat{\beta}_i) > 10$). In each iteration, we removed the feature with the highest VIF value and repeated this process until all VIF values were under the threshold of 10 (i.e., our stopping criteria).

$$\boldsymbol{VIF_i = VIF(\hat{\beta}_i) = \frac{1}{1 - R_i^{\,2}}} \quad (5)$$

After dealing with multicollinearity using our iterative procedure, the following features were excluded: *Partiality Score* and *Lot Type*. Also, the resulted model achieved lower scores compared to the above model. The resulted scores were: MAE score of 0.0446, MSE of 0.01, R^2 of 0.77, AIC of 41877.01, and BIC of 41720.5.



VIF VALUES

Starting Condition

```
                        variables          VIF
0                PARTIALITY_SCORE   122.364444
1                 OVERLAP_SECONDS     1.363866
2           PROCESSED_WAFER_COUNT    16.024173
3                      Wafer STDV     3.028129
4                        LOT_TYPE          inf
5                        LAST_LOT     1.350973
6        Lot calsification_1st Lot  187.019065
7   Lot calsification_1st Lot - MW   92.725596
8 Lot calsification_1st Lot - MW Pre   1.354240
9   Lot calsification_Embedded Lot    3.274919
10       Lot calsification_Mid Lot   97.394691
11  Lot calsification_Mid Lot - MW    4.909833
12 Lot calsification_Mid Lot - MW Pre  21.026996
13       Lot calsification_Nth Lot 1182.395875
14  Lot calsification_Nth Lot - MW  148.685661
15               Job_Non Prod job  242.490944
16                      Job_job 1          inf
17                     Job_job 10     2.195799
18                      Job_job 2          inf
19                      Job_job 3          inf
20                      Job_job 4          inf
21                      Job_job 5          inf
22                      Job_job 6          inf
23                      Job_job 7          inf
24                      Job_job 8          inf
25                      Job_job 9          inf
26               Machine_Machine1     1.859287
27               Machine_Machine2     1.638809
```

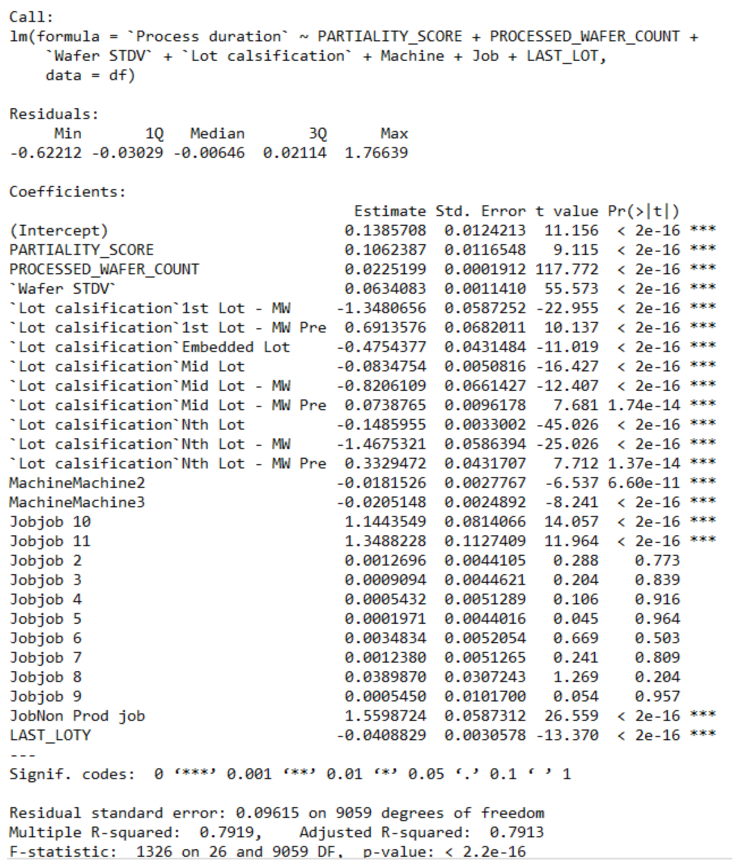
Ending Condition

```
                        variables       VIF
0                 OVERLAP_SECONDS  1.342570
1           PROCESSED_WAFER_COUNT  8.821415
2                      Wafer STDV  1.437824
3                        LAST_LOT  1.322223
4        Lot calsification_1st Lot  1.332660
5   Lot calsification_1st Lot - MW  1.223081
6 Lot calsification_1st Lot - MW Pre  1.004308
7   Lot calsification_Embedded Lot  1.012378
8        Lot calsification_Mid Lot  1.107093
9   Lot calsification_Mid Lot - MW  1.038042
10 Lot calsification_Mid Lot - MW Pre  1.042480
11  Lot calsification_Nth Lot - MW  1.207634
12                     Job_job 10  1.042871
13                      Job_job 2  2.938030
14                      Job_job 3  2.959179
15                      Job_job 4  1.521562
16                      Job_job 5  2.965367
17                      Job_job 6  1.481212
18                      Job_job 7  1.560822
19                      Job_job 8  1.005119
20                      Job_job 9  1.131798
21               Machine_Machine1  1.778763
22               Machine_Machine2  1.585508
```

Fig. 7. Handling multicollinearity by removing features with high VIF values. The initial state appears on the right, whereas the final state appears on the left.

Due to the loss in performance, we moved to the second strategy and used feature selection methods to see if they would result in a better model than the baseline model described earlier. We examined the following methods: *Forward Selection*, *Backward Elimination*, and *Stepwise*. As shown in **Fig. 8** below, we ended up with a unified model by applying all three methods. The resulted model excluded the following features: *Overlap Seconds* and *Lot Type*. Also, note that it had a slightly higher F-score of 1326 than the baseline score (i.e., 1227). Also, it achieved the same scores in terms of MAE, MSE, and R^2. However, since the number of features reduced from 9 to 7 features, it resulted in slightly lower AIC (-16742.181) and BIC (-16542.975) scores compared to the baseline model.

Moreover, we wanted to validate the model assumption using the appropriate methods. First, in order to check the linearity assumption, we visualized a residual plot to describe the relations between the errors, e_i and the predicted values, $\hat{y}_i$. As shown in **Fig. 10**, most of the residuals are distributed almost uniformly near the line of zero (i.e., a perfect prediction), and the trend is partially linear. However, from the dispersion perspective, i.e., variance, some predictions are slightly above the line (i.e., a positive residual) where the maximal residual equals 1.76, whereas in the opposite direction, some predictions were higher than the actual value resulted in a negative residual where the minimal was equal to -0.62. Since the variance tends to zero due to predictions' density, we can assume that the homoscedasticity assumptions take hold. However, to further investigate linearity and homoscedasticity, we plotted the standardized residuals, z_i, where $z_i = \frac{e_i - E(e_i)}{\sqrt{V(e_i)}}$, hence, centralized using the expectation of the error and scaled by the error's standard deviation. Based on the standardized residuals plot in **Fig. 9** below, we can see that the trend is linear for fitted values between 0.0 to 1.0 but afterward, the gradient increases and breaks the trend’s linearity. Therefore, we believe that applying transformations or removing anomalies has a higher moment that pulls the fitted model up can be an appropriate approach to fix this issue.



```
Call:
lm(formula = `Process duration` ~ PARTIALITY_SCORE + PROCESSED_WAFER_COUNT +
    `Wafer STDV` + `Lot calsification` + Machine + Job + LAST_LOT,
    data = df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.62212 -0.03029 -0.00646  0.02114  1.76639

Coefficients:
                                      Estimate Std. Error t value Pr(>|t|)
(Intercept)                          0.1385708  0.0124213  11.156  < 2e-16 ***
PARTIALITY_SCORE                     0.1062387  0.0116548   9.115  < 2e-16 ***
PROCESSED_WAFER_COUNT                0.0225199  0.0001912 117.772  < 2e-16 ***
`Wafer STDV`                         0.0634083  0.0011410  55.573  < 2e-16 ***
`Lot calsification`1st Lot - MW     -1.3480656  0.0587252 -22.955  < 2e-16 ***
`Lot calsification`1st Lot - MW Pre  0.6913576  0.0682011  10.137  < 2e-16 ***
`Lot calsification`Embedded Lot     -0.4754377  0.0431484 -11.019  < 2e-16 ***
`Lot calsification`Mid Lot          -0.0834754  0.0050816 -16.427  < 2e-16 ***
`Lot calsification`Mid Lot - MW     -0.8206109  0.0661427 -12.407  < 2e-16 ***
`Lot calsification`Mid Lot - MW Pre  0.0738765  0.0096178   7.681 1.74e-14 ***
`Lot calsification`Nth Lot          -0.1485955  0.0033002 -45.026  < 2e-16 ***
`Lot calsification`Nth Lot - MW     -1.4675321  0.0586394 -25.026  < 2e-16 ***
`Lot calsification`Nth Lot - MW Pre  0.3329472  0.0431707   7.712 1.37e-14 ***
MachineMachine2                     -0.0181526  0.0027767  -6.537 6.60e-11 ***
MachineMachine3                     -0.0205148  0.0024892  -8.241  < 2e-16 ***
Jobjob 10                            1.1443549  0.0814066  14.057  < 2e-16 ***
Jobjob 11                            1.3488228  0.1127409  11.964  < 2e-16 ***
Jobjob 2                             0.0012696  0.0044105   0.288    0.773
Jobjob 3                             0.0009094  0.0044621   0.204    0.839
Jobjob 4                             0.0005432  0.0051289   0.106    0.916
Jobjob 5                             0.0001971  0.0044016   0.045    0.964
Jobjob 6                             0.0034834  0.0052054   0.669    0.503
Jobjob 7                             0.0012380  0.0051265   0.241    0.809
Jobjob 8                             0.0389870  0.0307243   1.269    0.204
Jobjob 9                             0.0005450  0.0101700   0.054    0.957
JobNon Prod job                      1.5598724  0.0587312  26.559  < 2e-16 ***
LAST_LOTY                           -0.0408829  0.0030578 -13.370  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.09615 on 9059 degrees of freedom
Multiple R-squared:  0.7919,    Adjusted R-squared:  0.7913
F-statistic:  1326 on 26 and 9059 DF,  p-value: < 2.2e-16
```

Fig. 8. Output of the resulted linear regression model after applying all three feature selection methods.

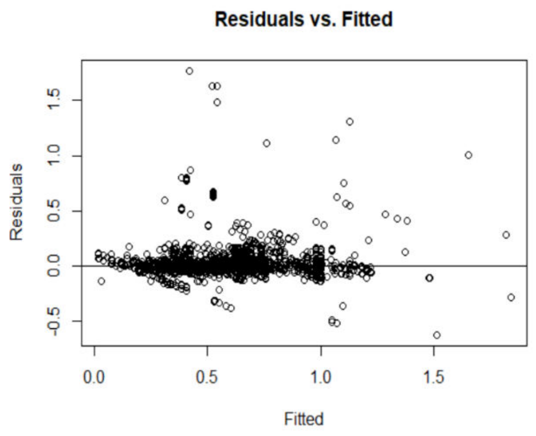


Fig. 10. Plot of residuals as function of fitted values based on the linear regression model resulted after forward selection.

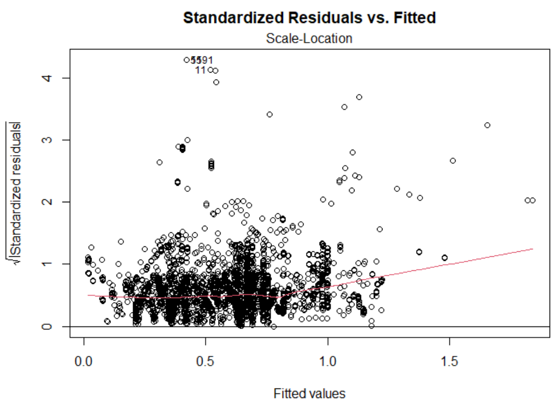
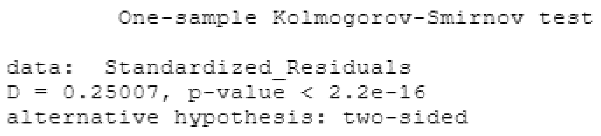


Fig. 9. Plot of standardized residuals as function of fitted values based on the linear regression model resulted after forward selection.

Second, to test if the normality assumption takes hold, we plotted a QQ-plot and conducted the Kolmogorov-Smirnov (KS) test (see **Fig. 11** below). **Fig. 12** below shows that most points do not approximately fall along the reference line. Thus, we can not assume normality. Further, the KS test resulted in a $p_{value}\ of\ 2.2e-16$. As a result, we have enough observations to reject the null hypothesis that the residuals were drawn from the normal distribution since the p_{value} is less than the significance level (i.e., $\alpha = 0.05$).



```
        One-sample Kolmogorov-Smirnov test

data:  Standardized_Residuals
D = 0.25007, p-value < 2.2e-16
alternative hypothesis: two-sided
```

Fig. 11 Kolmogorov-Smirnov test to test if normality assumptions takes hold.

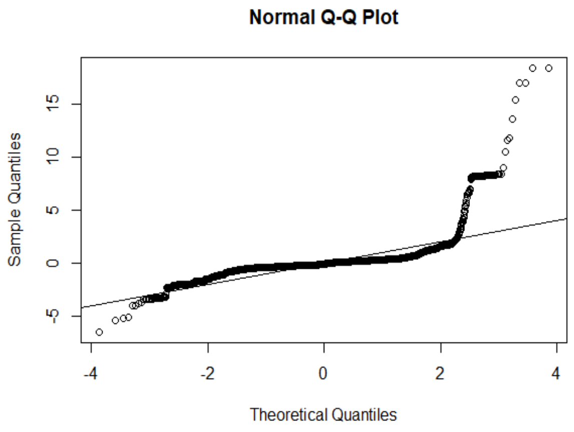
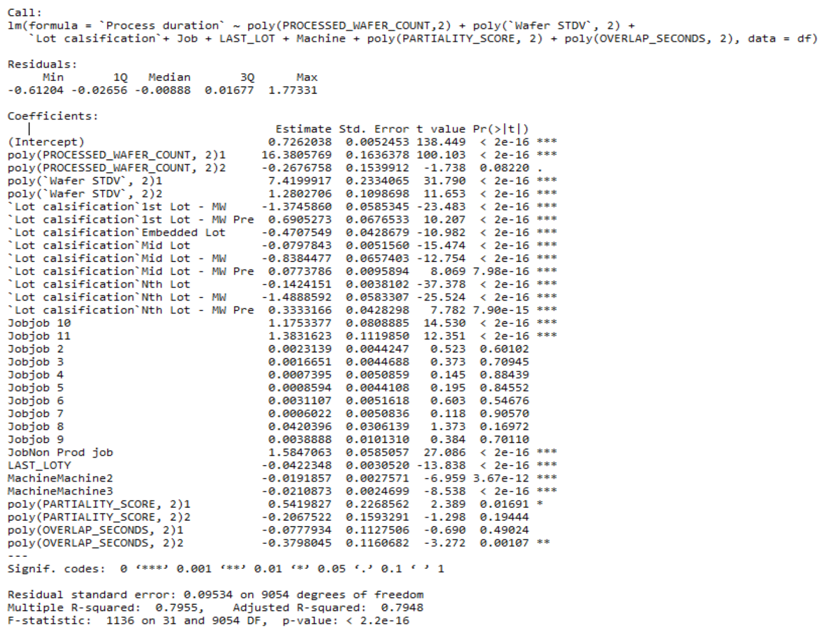


Fig. 12. QQ plot of residuals based on the linear regression model resulted after forward selection.

Furthermore, we thought that we should consider fitting a polynomial model to deal with the linearity issue. As a result, we used forward selection with a 2nd order polynomial transformation on the numeric features. We ended up with a slightly better model than the last mentioned model. In **Fig. 13** below, we can see that the resulted model achieved a F score of 1136, which was slightly lower than the model above F score of 1326. However, it was able to better explain the variance (~0.35 more than the linear model above) in terms of both R^2 and $R^2{}_{Adjusted}$, which were equal to 0.7955 and 0.7948 compared to the linear model scores, 0.792 and 0.791, respectively.



```
Call:
lm(formula = `Process duration` ~ poly(PROCESSED_WAFER_COUNT,2) + poly(`Wafer STDV`, 2) +
   `Lot calsification`+ Job + LAST_LOT + Machine + poly(PARTIALITY_SCORE, 2) + poly(OVERLAP_SECONDS, 2), data = df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.61204 -0.02656 -0.00888  0.01677  1.77331

Coefficients:
   |                                   Estimate Std. Error t value Pr(>|t|)
(Intercept)                           0.7262038  0.0052453 138.449  < 2e-16 ***
poly(PROCESSED_WAFER_COUNT, 2)1      16.3805769  0.1636378 100.103  < 2e-16 ***
poly(PROCESSED_WAFER_COUNT, 2)2      -0.2676758  0.1539912  -1.738  0.08220 .
poly(`Wafer STDV`, 2)1                7.4199917  0.2334065  31.790  < 2e-16 ***
poly(`Wafer STDV`, 2)2                1.2802706  0.1098698  11.653  < 2e-16 ***
`Lot calsification`1st Lot - MW      -1.3745860  0.0585345 -23.483  < 2e-16 ***
`Lot calsification`1st Lot - MW Pre   0.6905273  0.0676533  10.207  < 2e-16 ***
`Lot calsification`Embedded Lot      -0.4707549  0.0428679 -10.982  < 2e-16 ***
`Lot calsification`Mid Lot           -0.0797843  0.0051560 -15.474  < 2e-16 ***
`Lot calsification`Mid Lot - MW      -0.8384477  0.0657403 -12.754  < 2e-16 ***
`Lot calsification`Mid Lot - MW Pre   0.0773786  0.0095894   8.069 7.98e-16 ***
`Lot calsification`Nth Lot           -0.1424151  0.0038102 -37.378  < 2e-16 ***
`Lot calsification`Nth Lot - MW      -1.4888592  0.0583307 -25.524  < 2e-16 ***
`Lot calsification`Nth Lot - MW Pre   0.3333166  0.0428298   7.782 7.90e-15 ***
Jobjob 10                             1.1753377  0.0808885  14.530  < 2e-16 ***
Jobjob 11                             1.3831623  0.1119850  12.351  < 2e-16 ***
Jobjob 2                              0.0023139  0.0044247   0.523  0.60102
Jobjob 3                              0.0016651  0.0044688   0.373  0.70945
Jobjob 4                              0.0007395  0.0050859   0.145  0.88439
Jobjob 5                              0.0008594  0.0044108   0.195  0.84552
Jobjob 6                              0.0031107  0.0051618   0.603  0.54676
Jobjob 7                              0.0006022  0.0050836   0.118  0.90570
Jobjob 8                              0.0420396  0.0306139   1.373  0.16972
Jobjob 9                              0.0038888  0.0101310   0.384  0.70110
JobNon Prod job                       1.5847063  0.0585057  27.086  < 2e-16 ***
LAST_LOTY                            -0.0422348  0.0030520 -13.838  < 2e-16 ***
MachineMachine2                      -0.0191857  0.0027571  -6.959 3.67e-12 ***
MachineMachine3                      -0.0210873  0.0024699  -8.538  < 2e-16 ***
poly(PARTIALITY_SCORE, 2)1            0.5419827  0.2268562   2.389  0.01691 *
poly(PARTIALITY_SCORE, 2)2           -0.2067522  0.1593291  -1.298  0.19444
poly(OVERLAP_SECONDS, 2)1            -0.0777934  0.1127506  -0.690  0.49024
poly(OVERLAP_SECONDS, 2)2            -0.3798045  0.1160682  -3.272  0.00107 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.09534 on 9054 degrees of freedom
Multiple R-squared:  0.7955,    Adjusted R-squared:  0.7948
F-statistic:  1136 on 31 and 9054 DF,  p-value: < 2.2e-16
```

Fig. 13. Output of the resulted polynomial regression model after applying forward selection method.

In addition to the polynomial transformation mentioned above, we applied a box-cox transformation on the dependent variable (i.e., processing duration) by taking the log of it, that is, to address both normality and heteroscedasticity assumptions (see **Fig. 14** below). By doing so, as shown in **Fig. 15** below, the fitted model improved, and its resulted metrics were as follows: an MAE score of 0.081, MSE of 0.026, R^2 of 0.846, AIC of 7282.862, and BIC of 7076.545.

Further, as shown in **Fig. 14** below, the residuals' variance is spread uniformly along with the polynomial (red) line compared to the case described earlier in **Fig. 9** above.

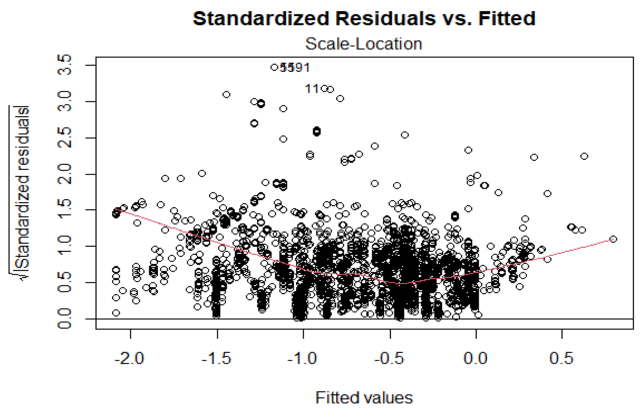
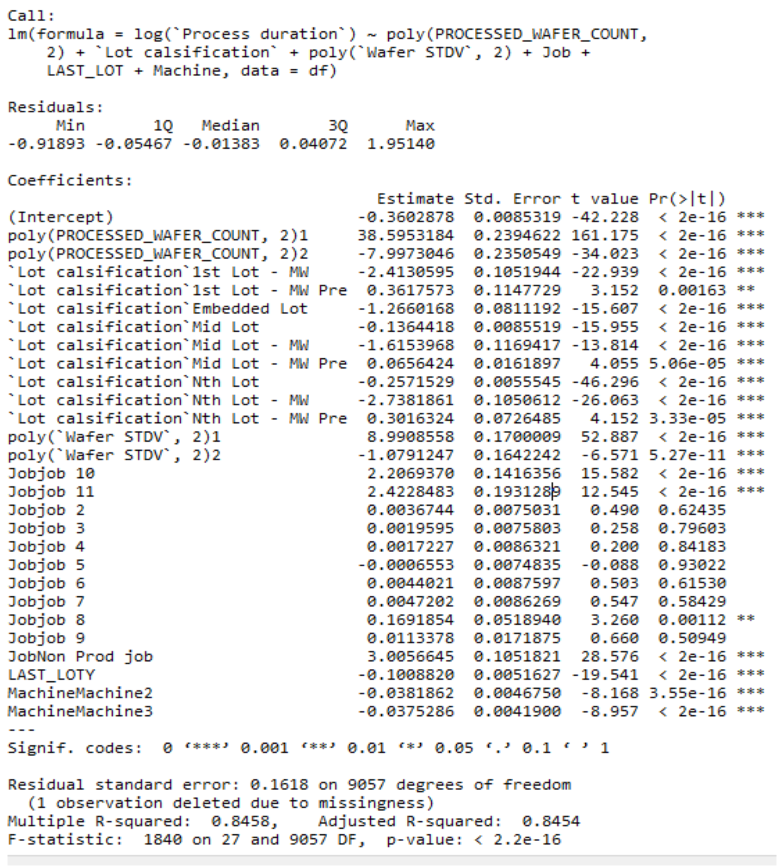


Fig. 14. Standardized residuals plot as function of fitted values based on the Box-Cox polynomial regression model resulted after forward selection.



```
Call:
lm(formula = log(`Process duration`) ~ poly(PROCESSED_WAFER_COUNT,
    2) + `Lot calsification` + poly(`Wafer STDV`, 2) + Job +
    LAST_LOT + Machine, data = df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.91893 -0.05467 -0.01383  0.04072  1.95140

Coefficients:
                                      Estimate Std. Error t value Pr(>|t|)
(Intercept)                         -0.3602878  0.0085319 -42.228  < 2e-16 ***
poly(PROCESSED_WAFER_COUNT, 2)1     38.5953184  0.2394622 161.175  < 2e-16 ***
poly(PROCESSED_WAFER_COUNT, 2)2     -7.9973046  0.2350549 -34.023  < 2e-16 ***
`Lot calsification`1st Lot - MW     -2.4130595  0.1051944 -22.939  < 2e-16 ***
`Lot calsification`1st Lot - MW Pre  0.3617573  0.1147729   3.152  0.00163 **
`Lot calsification`Embedded Lot     -1.2660168  0.0811192 -15.607  < 2e-16 ***
`Lot calsification`Mid Lot          -0.1364418  0.0085519 -15.955  < 2e-16 ***
`Lot calsification`Mid Lot - MW     -1.6153968  0.1169417 -13.814  < 2e-16 ***
`Lot calsification`Mid Lot - MW Pre  0.0656424  0.0161897   4.055 5.06e-05 ***
`Lot calsification`Nth Lot          -0.2571529  0.0055545 -46.296  < 2e-16 ***
`Lot calsification`Nth Lot - MW     -2.7381861  0.1050612 -26.063  < 2e-16 ***
`Lot calsification`Nth Lot - MW Pre  0.3016324  0.0726485   4.152 3.33e-05 ***
poly(`Wafer STDV`, 2)1               8.9908558  0.1700009  52.887  < 2e-16 ***
poly(`Wafer STDV`, 2)2              -1.0791247  0.1642242  -6.571 5.27e-11 ***
Jobjob 10                            2.2069370  0.1416356  15.582  < 2e-16 ***
Jobjob 11                            2.4228483  0.1931289  12.545  < 2e-16 ***
Jobjob 2                             0.0036744  0.0075031   0.490  0.62435
Jobjob 3                             0.0019595  0.0075803   0.258  0.79603
Jobjob 4                             0.0017227  0.0086321   0.200  0.84183
Jobjob 5                            -0.0006553  0.0074835  -0.088  0.93022
Jobjob 6                             0.0044021  0.0087597   0.503  0.61530
Jobjob 7                             0.0047202  0.0086269   0.547  0.58429
Jobjob 8                             0.1691854  0.0518940   3.260  0.00112 **
Jobjob 9                             0.0113378  0.0171875   0.660  0.50949
JobNon Prod job                      3.0056645  0.1051821  28.576  < 2e-16 ***
LAST_LOTY                           -0.1008820  0.0051627 -19.541  < 2e-16 ***
MachineMachine2                     -0.0381862  0.0046750  -8.168 3.55e-16 ***
MachineMachine3                     -0.0375286  0.0041900  -8.957  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1618 on 9057 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.8458,    Adjusted R-squared:  0.8454
F-statistic:  1840 on 27 and 9057 DF,  p-value: < 2.2e-16
```

Fig. 15. Output of the final regression model after applying Box-Cox transformation on process duration and 2nd order polynomial transformation on features selected using forward selection.

Finally, we ended up with the following independent features: *Processed Wafer Count* (1st & 2nd order polynomial), *Lot Calcification, Wafer Stdv* (1st & 2nd order polynomial), *Job, Last Lot,* and *Machine.* Also, note that since we used log transformation, in order to calculate the predicted process duration, we need to apply a reverse transformation using the exponent function.

3.4.1.3 *Support Vector Machine Regressor*

We tested the predictive power of the SVM regressor using the SVR class of the sklearn library. The SVR algorithm is similar to the SVM; however, rather than having the hyperplane act as a decision boundary in a classification problem, in SVR, a match is found between some vectors and a particular position on the hyperplane. We conducted this experiment using two types of kernels: *Linear* and *Radial Basis Function* (RBF). Also, note that for each kernel, we tuned the hyperparameter, C, a regularization parameter that controls the trade-off between gaining a low training error and a low testing error. Hence, C controls the ability to generalize to unseen data. For large C values, the algorithm will prefer a hyperplane with a smaller margin and smaller prediction errors to the train set. Conversely, lower values will create a hyperplane with a larger margin and larger prediction error. In our experiments, we tuned C for each integer value between 1 to 50.

3.4.1.3.1 Linear SVR

After optimizing the SVR model using the linear kernel, the best results were obtained using a $C = 20,$ and the resulted metrics were an MAE score of 0.105, MSE of 0.0245, R^2 of 0.445, AIC of 33678, and BIC of 33521.

3.4.1.3.2 Radial SVR

After optimizing the SVR model using the RBF kernel, the best results were obtained using a $C = 25,$ and the resulted metrics were an MAE score of 0.077, MSE of 0.018, R^2 of 0.582, AIC of 36291, and BIC of 36134.

As can be seen above, the SVR models obtained lower performance than the other models. Thus, this type of model is less suitable for the problem.

3.4.2 Non-parametric Models

When it comes to small-to-medium structured/tabular data, decision tree-based algorithms are considered appropriate candidates for both classification and regression tasks in the class of non-parametric models. There are two main ensemble-based techniques: *Bagging* and *Boosting*. Bagging or Bootstrap aggregating is an ensemble meta-algorithm combining multiple decision trees' predictions through a majority voting mechanism. In boosting, however, models are built sequentially by minimizing the errors from previous models while increasing (i.e., boosting) the influence of high-performing models.

In this study, we examined both techniques where Random Forest (RF) [4] is our chosen algorithm from the Bagging class, whereas XGBoost [5] is our chosen algorithm from the boosting class.

3.4.2.1 *Random Forest (Bagging)*

Random forest is a commonly-used machine learning algorithm that combines multiple decision trees to reach a single result. The random forest algorithm is an extension of the bagging method as it utilizes both bagging and feature randomness to create an uncorrelated forest of decision trees. Feature randomness generates a random subset of features, which ensures low correlation among decision trees. This is a crucial difference between decision trees and random forests. While decision trees consider all the possible feature splits, random forests only select a subset of those features.

We optimized the Random Forest regressor during our study by tuning several hyperparameters using Randomized Search 10-CV with 80 randomly selected combinations from a grid's parameters space of 540 combinations shown in **Table 2** below.

Table 2
Description of the Random Forest hyperparameters space.

ID	Hyperparameter	Description	Type	Examined Feature Space
1	bootstrap	method for sampling data points (with or without replacement)	Binary	{True, False}
2	max_depth	max number of levels in each decision tree	Integer	{60, 100, 150}
3	max_features	max number of features considered for splitting a node	Integer	{2, 3}
4	min_samples_leaf	min number of data points allowed in a leaf node	Integer	{2, 6, 8}
5	min_samples_split	min number of data points placed in a node before the node is split	Integer	{6, 10, 14}
6	n_estimators	number of trees in the foreset	Integer	{50, 200, 500, 1000, 2000}
	Total Feature Space (Combinations)			$2 \times 3 \times 2 \times 3 \times 3 \times 5 = 540$

After optimizing the RF model, the best results were obtained using the following combination of hyperparameters:

$$\{'\textbf{n\_estimators}': 200, \quad '\textbf{min\_samples\_split}': 14, \quad '\textbf{min\_samples\_leaf}': 2, \quad '\textbf{max\_features}': 3, \quad '\textbf{max\_depth}': 100, \quad '\textbf{bootstrap}': \text{False}, \}$$

As shown in **Fig. 16** below, we found that based on information gain, the most informative features are *Processed wafer count* (0.58), *Partiality Score* (0.13) and *Overlap Seconds* (0.13), and *Wafer Stdv* (0.08).

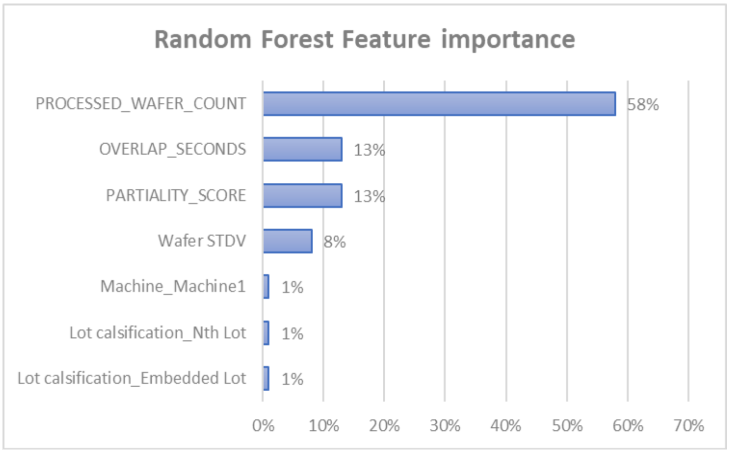


Fig. 16 . Random forest feature importance ranked by F-score in descending order.

3.4.2.2 *XGBoost (Boosting)*

XGBoost is an ML algorithm and a decision-tree-based ensemble that uses gradient boosting. Unlike Random Forest, which uses an ensemble bagging approach, XGBoost uses another approach called Boosting to create a collection of predictors. In this approach, learners are learned sequentially, with early learners fitting simple models to the data followed by error analysis. Consecutive trees (random sample) are fit, and at every step, the goal is to improve the accuracy from the prior tree. Also, note that XGBoost is suitable for both classification and regression tasks.

We optimized the XGBoost regressor during our study by tuning several hyperparameters using Randomized Search 10-CV with 1000 randomly selected combinations from a grid's parameters space of 165,375 combinations shown in **Table 3** below.

Table 3
Description of the XGBoost hyperparameters space.

ID	Hyperparameter	Description	Type	Examined Feature Space
1	Subsample	Subsample ratio of the training instance	Continuous	{0, 0.3, 0.6, 0.8, 1.0}
2	Num of Estimators	Number of gradient boosted trees	Integer	{10, 50, 100, 250, 400, 600, 1000}
3	Min Child Weight	Minimum sum of instance weight(hessian) needed in a child	Continuous	{1.0, 5.0, 10.0}
4	Max Depth	Maximum tree depth for base learners	Integer	{3, 4, 5, 6, 7, 8, 10, 12, 15}
5	Learning Rate	Boosting learning rate (η)	Continuous	{0.0001, 0.001, 0.002, 0.005, 0.01, 0.05, 0.1}
6	Gamma	Minimum loss reduction required to make a further partition on a leaf node of the tree	Continuous	{0, 0.5, 1, 2, 5}
7	Coloumns Sample by Tree	Subsample ratio of columns when constructing each tree	Continuous	{0, 0.3, 0.6, 0.8, 1.0}
	Total Feature Space (Combinations)			$5 \times 7 \times 3 \times 9 \times 7 \times 5 \times 5 = 165{,}375$

After optimizing the XGBoost model, the best results were obtained using the following combination of hyperparameters:

$$\{'\boldsymbol{subsample}': 0.8, \quad '\boldsymbol{n\_estimators}': 400, \quad '\boldsymbol{min\_child\_weight}': 1, \quad '\boldsymbol{max\_depth}': 4, \quad '\boldsymbol{learning\_rate}': 0.1, \quad '\boldsymbol{gamma}': 0, \quad '\boldsymbol{colsample\_bytree}': 0.8\}$$

The resulted metrics were an MAE score of 0.026, MSE of 0.003, R^2 of 0.934, AIC of 5360.4, and BIC of 5355.58. Further, as shown in **Fig. 17** on the right, the feature importance plot is displayed. The most informative features in terms of F-score are *Overlap Seconds* (1836), *Processed Wafer Count* (643), *Wafer Standard Deviation* (524), *Partiality Score* (313), and *Last Lot* (163).

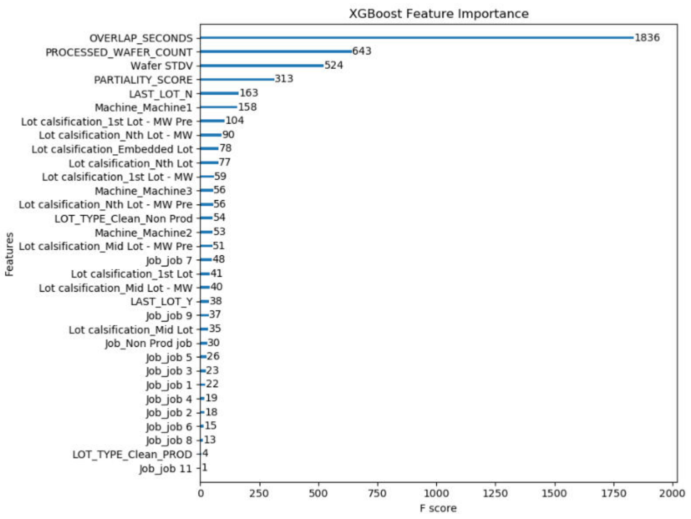


Fig. 17. XGBoost Feature importance ranked by F-score in descending order.

4. Results

As shown in **Table 4** below, in the family of parameter models, the linear regression model with all features outperformed the rest parametric models by all metrics MAE of 0.045, MSE of 0.01, R^2 of 0.821, AIC of 4452.84, and BIC of 4448.03. However, the polynomial regression with the Box-Cox transformation achieved the highest R^2 of 84.6%.

Further, under the family of non-parametric models, the XGBoost model outperformed the rest in terms of the lowest scores of MAE (0.026), MSE (0.003), and R^2 (93.4%). In addition, the standard deviation between folds is also the lowest, thus, implies that this model is more robust and stable in terms of generalization. However, Random Forest achieved the lowest results in terms of AIC (2471) and BIC (1900).

Table 4
Comparison between several ML-based models.

Model	**MAE**	**MSE**	$\boldsymbol{R^2}$	**AIC**	**BIC**
Linear Regression ($\lambda = 0$)	0.045 ($\pm$0.015)	0.01 ($\pm$0.01)	0.821 ($\pm$0.13)	4452.84 ($\pm$974.2)	4448.03 ($\pm$974.2)
Ridge Regression ($\lambda = 0.1$)	0.045 ($\pm$0.015)	0.01 ($\pm$0.01)	0.821 ($\pm$0.13)	(4690 $\pm$ 913.46)	4685.01 ($\pm$913.5)
Lasso Regression ($\lambda = 0.01$)	0.054 ($\pm$0.02)	0.013 ($\pm$0.014)	0.756 ($\pm$0.2)	(4453 $\pm$ 974.2)	4448.02 ($\pm$974.2)
Polynomial Regression (using Box-Cox)	0.081 ($\pm$0)	0.026 ($\pm$0)	0.846 ($\pm$0)	7282.86 ($\pm$0)	7076.54 ($\pm$0)
SVR (Linear)	0.105 ($\pm$0.005)	0.0245 ($\pm$0.003)	0.445 ($\pm$0.031)	33678 ($\pm$992)	33521 ($\pm$992)
SVR (RBF)	0.077 ($\pm$0.003)	0.018 ($\pm$0.0026)	0.5822 ($\pm$0.046)	36291 ($\pm$1332)	36134 ($\pm$1332)
Random Forest	0.053 ($\pm$0.002)	0.0098 ($\pm$0.002)	0.779 ($\pm$0.04)	2471 ($\pm$199.6)	1900 ($\pm$199.98)
XGBoost	0.026 ($\pm$0.001)	0.003 ($\pm$0.001)	0.934 ($\pm$0.026)	5360.39 ($\pm$346.3)	5355.58 ($\pm$346.3)

5. Summary and Conclusions

In conclusion, we answered our research question by developing models that can accurately predict the processing duration based on various features collected from the production's machines' logs. The best models chosen are the XGBoost with an average error of up to 1.5 minutes relative to the actual process duration (i.e., target variable), i.e., less than 4.3% relative to the total processing time (36.5 minutes). The second model that stood out in the AIC and BIC indices is the random forest with an average error of 3.2 minutes relative to the actual processing time, i.e., three times the previous model. However, since each model has its advantages, we recommend that in the future, consider using an ensemble approach between the models and examine whether the indices improve.

We compared the most informative features (i.e., important features) chose by our selected models. It seems that the most influential features are *Processed Wafer Count* (i.e., the number of wafers per lot) and *Wafer Stdv* (i.e., the difference between the number of wafers and the processing units). That is, if we want to improve and decrease process duration without compromising productivity, we need to find ways to minimize the different processing in the way the wafers are divided between the different processing units. To meet this goal, one can conduct future research and go down to a lower resolution to understand the technological root factors that caused the differences in the division.

Other essential features that have appeared in at least two models are the number of processing units on which at least one wafer runs (i.e., *Partiality Score*), the overlap time of two wafers lots simultaneously on a machine (i.e., *Overlap Seconds*), and an indicator variable as to whether the wafers lot is the last in the sequence (i.e., *Last Lot*).

Examining the correlations between variables identified the same features listed above when the *Processed Wafer Count* feature was positively correlated with almost 70% with the processing time (*i.e., Process Duration*), which indicates why it was chosen among all the different models. The Wafer Stdv feature had a nearly 50% positive correlation with processing time and was also selected by all models.

Also, we emphasize that according to the correlation matrix, it can be seen that if the number of active processing units decrease (i.e., lower *Partiality Score*), the variance in the distribution of wafers on processing units increase (i.e., higher *Wafer Stdv*). That is, as a result of the negative correlation between these features (-0.62). Therefore, due to transitivity, *Process Duration* also increases due to the positive correlation between *Wafer Stdv* and *Process Duration* (0.48).

6. Reference

[1] H. L. SEAL, "Studies in the History of Probability and Statistics. XV The historical development of the Gauss linear model," *Biometrika*, vol. 54, no. 1–2, pp. 1–24, Jun. 1967.

[2] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[3] S. Abe and K. Onishi, "Sparse least squares support vector regressors trained in the reduced empirical feature space," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2007, vol. 4669 LNCS, no. PART 2, pp. 527–536.

[4] T. K. Ho, "Random decision forests," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 1995, vol. 1, pp. 278–282.

[5] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System."

[6] P. Vassiliadis and A. Simitsis, "EXTRACTION, TRANSFORMATION, AND LOADING."

[7] T. M. Cover, "LEARNING IN PATTERN RECOGNITION," in *Methodologies of Pattern Recognition*, Elsevier, 1969, pp. 111–132.

[8] F. & Ponce *et al.*, *Artificial Intelligence A Modern Approach Fourth Edition*. 2021.

[9] A. E. Hoerl and R. W. Kennard, "Ridge Regression: Biased Estimation for Nonorthogonal Problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[10] R. Tibshirani, "Regression Shrinkage and Selection Via the Lasso," *J. R. Stat. Soc. Ser. B*, vol. 58, no. 1, pp. 267–288, Jan. 1996.