A NEW METHOD FOR DESIGN PARAMETER OPTIMIZATION OF PRODUCTS OR PROCESSES WITH AN ORDINAL CATEGORICAL RESPONSE

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

PINAR ERDOĞAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
INDUSTRIAL ENGINEERING

AUGUST 2022

Approval of the thesis:

A NEW METHOD FOR DESIGN PARAMETER OPTIMIZATION OF PRODUCTS OR PROCESSES WITH AN ORDINAL CATEGORICAL RESPONSE

submitted by **PINAR ERDOĞAN** in partial fulfillment of the requirements for the degree of **Master of Science** i**n Industrial Engineering, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Esra Karakasal
Head of the Department, **Industrial Engineering**

Prof. Dr. Gülser Köksal
Supervisor, **Industrial Engineering, METU**

Assist. Prof. Dr. Leman Esra Dolgun
Co-Supervisor, **Industrial Engineering, ESTU**

Examining Committee Members:

Prof. Dr. Pelin Bayındır
Industrial Engineering, METU

Prof. Dr. Gülser Köksal
Industrial Engineering, METU

Assist. Prof. Dr. Leman Esra Dolgun
Industrial Engineering, ESTU

Prof. Dr. Yasemin Serin
Industrial Engineering, METU

Prof. Dr. Sinan Gürel
Industrial Engineering, METU

Date: 29.08.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name Last name: Pınar Erdoğan

Signature:

ABSTRACT

A NEW METHOD FOR DESIGN PARAMETER OPTIMIZATION OF PRODUCTS OR PROCESSES WITH AN ORDINAL CATEGORICAL RESPONSE

Erdoğan, Pınar
Master of Science, Industrial Engineering
Supervisor: Prof. Dr. Gülser Köksal
Co-Supervisor: Assist. Prof. Dr. Leman Esra Dolgun

August 2022, 226 pages

It is an important design problem to obtain the best parameter values that are insensitive to the variability of input, process or environmental factors of a product or process. Therefore, many industrial organizations use robust parameter design (RPD). While there are many methods in the literature for design parameter optimization with continuous quality characteristics, fewer studies are found for ordered categorical quality characteristics. This study proposes a new method for finding robust levels of design parameters of products and processes with ordinal categorical responses. Many approaches use expected value and variance as measures of location and dispersion respectively when the response is categorical. However, these measures used for numerical data are not meaningful to summarize categorical data. The developed method uses the median value and coefficient of ordinal variation (COV) for ordinal categorical data as location and dispersion measures, respectively. In addition, the Extreme Gradient Boosting (XGBoost) algorithm is presented as an alternative to Random Forests and Logistic Regression, which have been used in RPD studies in the literature. The classification performances of these algorithms are compared with Multi-Objective Decision Making methods. Based on this comparison, Random Forests algorithm is used to predict the category probabilities. These probabilities are used to calculate median and COV. Ordinal Logistic Regression and least squares regression are used to model these measures as functions of design parameters. The best parameter values are determined by solving a non-linear optimization problem. The proposed method is applied to different problems, and the results are discussed.

Keywords: Robust Parameter Design, Coefficient of Ordinal Variation (COV), Median, XGBoost, Random Forest

ÖZ

SIRALI KATEGORİK YANITA SAHİP ÜRÜN VE SÜREÇLERİN TASARIM PARAMETRESİ EN İYİLEMESİ İÇİN YENİ BİR METOT

Erdoğan, Pınar
Yüksek Lisans, Endüstri Mühendisliği
Tez Yöneticisi: Prof. Dr. Gülser Köksal
Ortak Tez Yöneticisi: Dr. Öğr. Üyesi Leman Esra Dolgun

Ağustos 2022, 226 sayfa

Bir ürün veya sürecin girdi, süreç veya çevre faktörlerindeki değişkenliğe karşı duyarsız en iyi parametre değerlerini elde etmek önemli bir tasarım problemidir. Bu nedenle, birçok endüstriyel organizasyon robust parametre tasarımını kullanmaktadır. Sürekli yanıta sahip kalite karakteristiklerinin tasarım parametre eniyilemesi için litertürde birçok yöntem bulunurken, sıralı kategorik yanıta sahip kalite karakteristikleri için daha az çalışmaya rastlanmaktadır. Bu çalışma, sıralı kategorik yanıt veren ürün ve süreçlerin tasarım parametrelerinin robust seviyelerinin bulunması için yeni bir yöntem önermektedir. Birçok yaklaşım, yanıt kategorik olduğunda konum ve dağılım ölçüsü olarak beklenen değer ve varyans kullanmaktadır. Ancak, numerik veriler için kullanılan bu ölçüler kategorik verileri özetlemek için anlamlı değildir. Geliştirilen yöntem, sıralı kategorik veriler için önerilmiş olan ortanca değer ve sıralı varyasyon katsayısını (COV) sırasıyla konum ve dağılım ölçüsü olarak kullanmaktadır. Ayrıca, Extreme Gradient Boosting (XGBoost) algoritması, literatürde bu alanda daha önce kullanılmış olan Rassal Ormanlar ve Lojistik Regresyona alternatif olarak sunulmuştur. Bu algoritmaların sınıflandırma performansı Çok amaçlı Karar Verme yöntemleri kullanılarak karşılaştırılmıştır. Belirli parametre değerlerinde yanıtın ortanca değer ve COV ölçülerini modellemek için Rassal Ormanlar algoritması kullanılmıştır. En iyi parametre değerleri bu iki ölçü için doğrusal olmayan optimizasyon probleminin çözülmesi ile elde edilir. Önerilen yöntem farklı problemlere uygulanmıştır, sonuçlar tartışılmıştır.

Anahtar Kelimeler: Robust Parametre Tasarımı, Sıralı Varyasyon Katsayısı (COV), Ortanca Değer, XGBoost, Rassal Ormanlar

To my family

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my supervisor Prof. Dr. Gülser Köksal and co-supervisor Assist. Prof. Dr. Leman Esra Dolgun for their advice, criticism, precious guidance and insight throughout the research. I am also grateful to them for their generosity on sharing their knowledge and experience during this study.

I would like to express my deepest thanks to my mother Kamuran Erdoğan, my father Ahmet Serhat Erdoğan for their invaluable support, endless understanding and patience.

TABLE OF CONTENTS

CHAPTERS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

It is a well-known fact that high quality products and services are closely related to reducing costs by reducing waste and at the same time increasing customer satisfaction. For this reason, it is important to produce high quality products or processes that are insensitive to variation. Robust parameter design (RPD) is a method developed to minimize variation caused by design and to achieve the best quality level.

In RPD studies, design parameters are optimized so that the system becomes insensitive to sources of variation and at the same time achieves the desired quality. There are two types of design parameters which are input parameters or factors of the products or processes: controllable and uncontrollable (noise) factors. In this approach, controllable factors (design parameters) are selected so that the effect of noise variables becomes minimum. So, it is important to analyze both location and dispersion effect of controllable factors to adjust the mean of the response and minimize the variance. The quality characteristic (quality level, response) can be defined as either continuous or categorical. It is relatively easy to analyze continuous quality characteristics, and there are many methods that have been proposed for this purpose. However, in many real-life data, quality characteristics are measured as categorical. Therefore, it is important to analyze categorical data properly to provide a better understanding of response variables.

All RPD studies involving the categorical response, use expected value and variance, which are mainly used to analyze continuous data, to see the effect of location and dispersion. Main contribution of this study is proposing a new method that uses median and COV, which are better to explain location and dispersion of ordered categorical (ordinal) data. This method can be applied to the problems where response is naturally ordered such as bad / acceptable / good / excellent.

In this thesis, dispersion and location measures are analyzed in terms of their values obtained for different probability distributions and correlation between them. Based on the literature review and this analysis, it is decided to use median and coefficient of ordinal variation (COV) as location and dispersion measures respectively in the proposed RPD with ordinal response method. These measures are based on category probabilities. Three methods as Ordinal Logistic Regression, Random Forest and XGBoost are considered for prediction of category probabilities. Once their parameters are tuned for four case studies, their performances, measured in terms of three criteria: precision, recall and area under curve (AUC), are compared using multiple criteria decision making (MCDM) methods TOPSIS and Multi-MOORA. A sensitivity analysis is also performed for the weights used in TOPSIS. Based on these analyses and by considering easiness in implementation, Random Forest is selected as the method to predict the category probabilities. The proposed method for RPD with ordinal response assumes that the data is collected according to an experimental design and predicts category probabilities at each experimental trial using RF and calculates median and COV at each experimental trial using these probabilities. Empirical models of the median and COV are built as functions of design parameters. Using these empirical models, a nonlinear optimization model is solved to find the levels of design parameters that produce desired median value and minimize the COV. The proposed method applied three case studies, one of them is smaller-the-better type and two of them are larger-the-better type of problems.

The subsequent sections are organized as follows. The related literature review for RPD with an ordinal response, measure selection for ordered categorical data, data mining algorithms and MCDM methods are given in Section 2. The comparison of the data mining algorithms (Ordinal Logistic Regression, Random Forest and XGBoost), proposed method for RPD with ordinal response and application of the proposed method to three case studies are shared in Section 3. The advantages and disadvantages of the method are discussed in Section 4. Finally, some conclusions and future research areas are proposed in Section 5.

CHAPTER 2

LITERATURE REVIEW

The experiments conducted in the early production stages can improve the quality of a product or process significantly and help to decrease cost and waste (Logothetis, 1992). Therefore, improving the quality of product or process in the pre-production stage (off-line) is important to reduce product development and lifetime costs and to increase product manufacturability and reliability (Kackar, 1985).

This section is organized as follows. First, brief literature review of existing RPD with an ordinal response methods is given. Then, location and dispersion measures suggested for ordinal categorical data are explained and an overview of ensemble techniques such as XGBoost and Random Forest is provided. The background of MCDM methods is given as well.

2.1 Robust Parameter Design with an Ordinal Response

RPD is a technique that reduces the sensitivity of an engineering design to the sources of variation so that high-quality products or services can be produced quickly, consistently and at a low cost (Kackar, 1985; Phadke, 1989). There are two types of input in RPD: controllable factors (design parameters) and noise factors (uncontrollable factors). Control factors can be easily controlled by the system, but noise factors can be difficult to control or costly to control. RPD involves choosing optimal parameter setting to find an optimal response. The ultimate aim in the RPD approach is making a product or process insensitive to noise factors while keeping the quality at the target level (Robinson et al., 2004).

The response in an RPD problem can be continuous or categorical (can be binary, ordinal or nominal). RPD methods with continuous response have been studied by many researchers. However, when response is categorical there are few studies in the literature. In ordinal categorical response cases response is categorized using natural order such as: bad / fair / good / excellent. The problem type with ordinal response can be smaller-the better where the best quality level is the lowest category, or larger-the-better where the target quality level is the highest category. There are many real-life problems with ordinal or nominal responses.

Taguchi (1974) proposes a method called Accumulation Analysis (AA) to analyze categorical response based on cumulative frequencies and then analysis of variance (ANOVA) is applied. The complexity of the procedure and its deficiency to not examine location and dispersion effects separately is criticized by many researchers (Box & Jones, 1986; Nair, 1986). Also, AA method is appropriate for the data with categorical controllable factors or the data that contains both categorical-continuous factors (Erdural, 2006; Gülbudak Dil, 2018; Logothetis, 1992). Scoring Scheme method is suggested by Nair (1986). This method calculates location and dispersion effects separately for parameter settings. Scores are given to each category for both location and dispersion and then ANOVA is applied. The combination of General Linear Model GLM) and Bayesian estimation techniques is used by Chipman and Hamada (1996) to analyze ordered categorical data. This method also uses uncertainty in the parameter estimation. However, Erişkin et al. (2021) mentions that one disadvantage of this method is the computational complexity compared to other methods.

The other contributions to the RPD with ordered categorical response are: Weighted-Signal-to-Noise Ratio (WSNR) is developed by Taguchi, and then, Wu and Yeh (2006) introduces and compares it with four different robust parameter design methods. Like AA, WSNR does not examine location and dispersion effects separately. Weighted Probability Scoring Scheme (WPSS) is developed by Jeng and

Guo (1996) and Minimization of Expected Loss (MEL) is introduced by Asiabar and Ghomi (2007).

Erdural (2006) proposes a method called Logistic Regression Model Optimization (LRMO). Erdural (2006) uses Logistic Regression to estimate class (category) probabilities of each experimental point using the logit link function and then calculates expected value and variance. Then, Signal to Noise Ratio (SNR) is calculated for each experimental point. ANOVA is applied to find the levels of the design parameters that maximize the SNR. This method is applicable for both smaller-the-better and larger-the-better types of problems. Köksal et al. (2006) mentions that LRMO method is an easy and effective way to find optimal parameter setting for both continuous and categorical controllable factors.

Karabulut (2013) compares different RPD methods such as: LRMO, AA, WSNR, SS, WPSS in her M. Sc. thesis study. It is concluded that LRMO and AA show better performance among the Logistic Regression dependent methods and ANOVA based methods, respectively.

Gülbudak Dil (2018) introduces a new method based on Random Forest in her M.Sc. thesis study. Random Forest is used instead of Logistic Regression to analyze the data and obtain probabilities. Once the class probabilities are estimated, expected value and variance of each experimental point are calculated, regression models that relate expected value and variance with the design parameters are built and ε-constraint method is applied to find the best parameter setting with minimum variance and target quality level.

Lastly, Erişkin et al. (2021) proposes design parameter continuous optimization for the ordinal response (DPCOOR). The proposed method uses Ordinal Logistic Regression to estimate probability of observing each category as a function of control factors where logit link is used as the link function as shown in Equations (2.1) and (2.2). These probabilities are used to express location (expected value) and dispersion (variance) as functions of control factors as given in Equation (2.3) and Equation (2.4), respectively.

$$\hat{P}(Y(x) \leq i) = \frac{exp\ (\hat{\gamma_i} + \hat{\beta}x)}{1 + exp\ (\hat{\gamma_i} + \hat{\beta}x)}\ , i = 1,2, \ldots, m - 1 \tag{2.1}$$

$$\hat{P}(Y(x) = i) = \hat{P}(Y(x) \leq i) - \hat{P}(Y(x) \leq i - 1), i = 1,2, \ldots, m \tag{2.2}$$

where m is the number of categories for the response.

$$\hat{E}\big(Y(x)\big) = \sum_{i=1}^{m} i(\hat{P}(Y(x) = i)) \tag{2.3}$$

$$\hat{V}\big(Y(x)\big) = \hat{E}[(Y)^2] - \widehat{[E}(Y)]^2 = \sum_{i=1}^{m} i^2 \left(\hat{P}(Y(x) = i)\right) - \hat{E}[(Y(x))]^2 \tag{2.4}$$

The purpose of the method is finding the levels of the control factors that maximize (if the response is LTB type) or minimize (if the response is STB type) estimated expected value given in Equation (2.3) and minimize the estimated variance given in Equation (2.4). Erişkin et. al (2021) uses desirability function method to find optimal parameter design. DPCOOR is also useful for experimental datasets which have categorical and continuous controllable factors. Another advantage of DPCOOR is, it can analyze continuous data directly with Logistic Regression and as a result, it can generate better parameter levels for continuous factors than other methods in the literature. Erişkin et al. (2021) mentions that instead of expected value and variance, using appropriate measures for ordinal data can be tried.

None of the methods proposed for RPD with ordinal categorical response explained in the previous paragraphs use location and dispersion metrics defined for ordinal data, instead mainly expected value and variance are selected for this purpose. The aim of this study is to propose a new method to analyze ordinal response with appropriate metrics.

2.2 Selection of Location and Dispersion Measures for Ordinal Data

There are two different opinions for analyzing data for ordinal categorical data. The first approach follows the idea that applying arithmetic calculations to ordinal data would be inappropriate because although the category numbers express an order, these are just arbitrary numbers. Instead of using mean and variance, which are meaningful location and dispersion measures for interval or ratio scale data, using median for ordinal and mode for nominal data as location measures and specifically designed measures for dispersion would be more meaningful to analyzing categorical data (Stevens, 1946). For this approach, proposed measures are mostly based on probabilities and cumulative probabilities of the categories. On the other hand, in the second approach categorical data is analyzed like continuous (interval or ratio scale) data, therefore mean and variance are used to calculate location and dispersion respectively (Allaj, 2018; Weisberg, 1992). In this approach integer scoring (or rank counts system) is adopted by researchers, assigning 1 to the first category and 2 to the second category, and so on.

Before starting to explain the related literature, let us introduce the notation used in the rest of the thesis for the sake of simplicity. Let Y is the ordinal categorical variable with m categories. The probability mass function is $p_i = P(Y = i)$, $i = 1,2, \ldots, m$ or the vector of $\boldsymbol{p} = (p_1, p_2, . . , p_m)'$ and the cumulative distribution function is the vector of $\boldsymbol{f} = (f_1, f_2, \ldots, f_m)'$. The two important cases for analyzing ordinal dispersion are: one point (the degenerate) distribution $\boldsymbol{f_{one}} = (0,1, \ldots,1)'$ and the polarized distribution where the probabilities are distributed evenly on the minimum and maximum categories such as: $p_1 = p_m = 0.5$ and $p_i = 0$ otherwise. For ordinal data, minimum and maximum dispersions are associated with the one-point distribution and polarized distribution, respectively (Kvålseth, 2011; Weiß, 2019a).

Leik's ordinal variation (LOV) is the first dispersion measure proposed for ordinal categorical data (Kvålseth, 2011) which is calculated as given in Equation (2.5). LOV $= 1$ indicates maximal dispersion and LOV $= 0$ indicates no dispersion at all.

$$\mathrm{LOV} = 1 - \frac{\sum_{i=1}^{m-1} |2f_i - 1|}{m-1} \tag{2.5}$$

Then Kvålseth (1995a) proposes Δ for ordinal dispersion.

$$\Delta = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} |i-j|\, p_i p_j \tag{2.6}$$

where $|i - j|$ is the absolute value of difference between category numbers i and j and by normalizing Δ with the maximum category number, ordinal variation in [0, 1] scale is obtained.

$$\Delta^{*} = \frac{\Delta}{(m - 1/4)} = \frac{2}{m-1} \sum_{i=1}^{m} \sum_{j=1}^{m} |i-j| p_i p_j \tag{2.7}$$

Kvålseth (1995a) also mentions that Δ^{*} equals to the index of ordinal variation (IOV) suggested by (Berry & Mielke, 1992a; 1992b) when sample size, N, is even and when N is odd it is $\Delta^{*} = (1 - 1/N^2)$IOV. The two drawbacks of Δ^{*} according to Kvålseth (1995a) are: first Δ^{*} can change in range $[0, 1]$ which is unreasonably and misleadingly large. Second, since Δ^{*} uses category numbers, although the category numbers show the rank or score, these are just arbitrary numbers. On the other hand, Blair & Lacy (1996) explains that $|i - j|$ does not show the distance between categories, it is just counting the scores between categories.

In the same study Kvålseth proposes coefficient of ordinal variation (COV). For the one-point distribution case COV is 0, and it is 1 for the polarized distribution.

$$\mathrm{COV} = 1 - \sqrt{1 - \Delta^{*}} \tag{2.8}$$

Kvålseth (1995b) uses the cumulative probabilities for calculating the Δ which is free from limitations of scoring system. As a results, Δ and COV also can be calculated as follows.

$$\Delta = \sum_{i=1}^{m-1} f_i(1 - f_i) \tag{2.9}$$

$$\mathrm{COV} = 1 - \sqrt{\frac{\sum_{i=1}^{m-1} |2f_i - 1|^2}{m - 1}} \tag{2.10}$$

Blair and Lacy (1996) develops a new measure, LSQ, and explains the relationship between LSQ and LOV, IOV and COV. They treat cumulative probability of the ordinal data as $m - 1$-dimensional array and this way the ordinal random variable is converted a point in Euclidean f-space, $(f_1, f_2, \dots, f_{m-1})$. LSQ is also range between $[0, 1]$. LSQ$= 0$ shows the maximal dispersion, and LSQ $= 1$ shows no-dispersion. In this respect, LSQ evaluates dispersion differently from the other measures. For the cases where sample size is even, LSQ is calculated as given in Equation (2.11).

$$\mathrm{LSQ} = \frac{\sum_{i=1}^{m-1} (f_i - 0.5)^2}{(m - 1)/4} \tag{2.11}$$

Blair and Lacy (1996) also shows that, the difference between LSQ and LOV is, LOV uses city-block metric for measuring the distance between categories. They mentions that $\mathrm{IOV} = 1 - \mathrm{LSQ}$ and $\mathrm{COV} = 1 - (1 - \mathrm{IOV})^{\frac{1}{2}}$, therefore, $\mathrm{COV} = 1 -$ LSQ. For all cases, even when sample size is odd, they suggests measure of ordinal dispersion (MD). This is the simplified version of LSQ.

$$\mathrm{MD} = \frac{D - D_{min}}{D_{max} - D_{min}} \tag{2.12}$$

where D is the distance between observed point and polarized distribution. D_{min} is the minimum distance to the polarized distribution and D_{max} is the maximum distance to the polarized distribution.

Kvålseth (2011) introduced a family of ordinal variation measures using Minkowski metric distance of $q \geq 1$ as shown in Equation (2.13).

$$\mathrm{OV}_q = 1 - \left(\frac{\sum_{i=1}^{m-1} |2f_i - 1|^q}{m-1}\right)^{\frac{1}{q}} \tag{2.13}$$

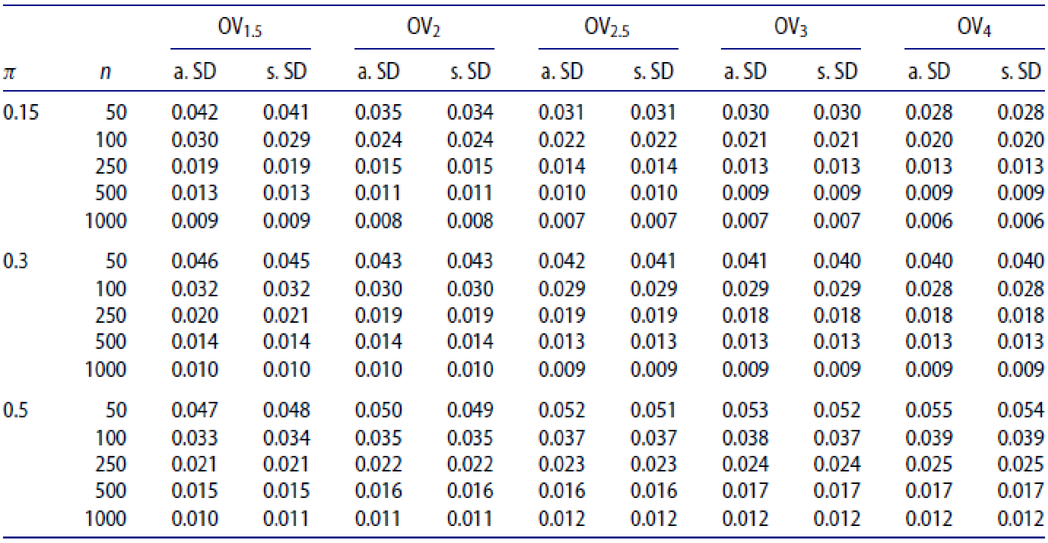
In that case, LOV $= \mathrm{OV}_1$ and COV $= \mathrm{OV}_2$, so LOV and COV uses city-block distance and Euclidean distance, respectively. Similar to LOV and COV other members of OV_q family range between $[0, 1]$.

Weiß (2019a) examines the asymptotic distribution of $\widehat{OV}_q$ family and evaluates the results of approximations with simulations. While deriving the approximations, $q = 1$ is separated from others because of the discontinuity at $f_i = \frac{1}{2}$.

In the simulation study, $i.i.d.$ ordinal data is generated by binomially distributed rank count $I \sim Bin(m,\ \pi)$ where π corresponds to the strongly asymmetric, asymmetric, and symmetric cases, $0.15, 0.3, 0.5$, respectively.

Table 2.1 shows the comparison of approximate and simulated standard deviations. Even for the smaller sample sizes such as $n = 50$, there is nearly perfect agreement between the asymptotic standard deviations and simulated standard deviations for all values of q.

Table 2.1 Ordinal data $x_1, \dots, x_m$ with $i.i.d.$ rank counts $I, I \sim Bin(m,\ \pi)$; asymptotic vs. simulated standard deviation (a. SD vs. s. SD) of $\widehat{OV}_q$ (Source: Weiß (2019a))

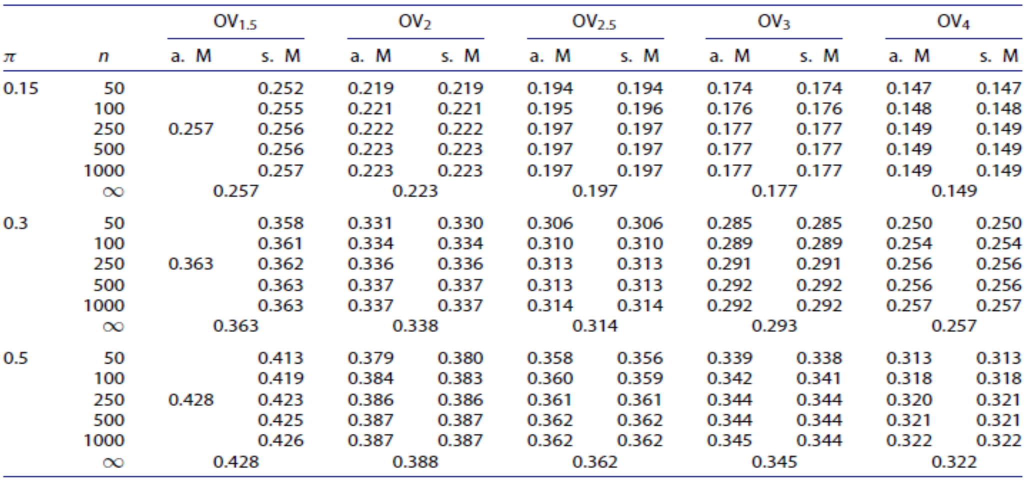


		OV$_{1.5}$		OV$_2$		OV$_{2.5}$		OV$_3$		OV$_4$	
π	n	a. SD	s. SD	a. SD	s. SD	a. SD	s. SD	a. SD	s. SD	a. SD	s. SD
0.15	50	0.042	0.041	0.035	0.034	0.031	0.031	0.030	0.030	0.028	0.028
	100	0.030	0.029	0.024	0.024	0.022	0.022	0.021	0.021	0.020	0.020
	250	0.019	0.019	0.015	0.015	0.014	0.014	0.013	0.013	0.013	0.013
	500	0.013	0.013	0.011	0.011	0.010	0.010	0.009	0.009	0.009	0.009
	1000	0.009	0.009	0.008	0.008	0.007	0.007	0.007	0.007	0.006	0.006
0.3	50	0.046	0.045	0.043	0.043	0.042	0.041	0.041	0.040	0.040	0.040
	100	0.032	0.032	0.030	0.030	0.029	0.029	0.029	0.029	0.028	0.028
	250	0.020	0.021	0.019	0.019	0.019	0.019	0.018	0.018	0.018	0.018
	500	0.014	0.014	0.014	0.014	0.013	0.013	0.013	0.013	0.013	0.013
	1000	0.010	0.010	0.010	0.010	0.009	0.009	0.009	0.009	0.009	0.009
0.5	50	0.047	0.048	0.050	0.049	0.052	0.051	0.053	0.052	0.055	0.054
	100	0.033	0.034	0.035	0.035	0.037	0.037	0.038	0.037	0.039	0.039
	250	0.021	0.021	0.022	0.022	0.023	0.023	0.024	0.024	0.025	0.025
	500	0.015	0.015	0.016	0.016	0.016	0.016	0.017	0.017	0.017	0.017
	1000	0.010	0.011	0.011	0.011	0.012	0.012	0.012	0.012	0.012	0.012

Similarly, asymptotic and simulated means are compared in Table 2.2, but this time in addition to the sample sizes given, $n = \infty$ is also included in the study. When true value of OV_q, $n = \infty$, and simulated means are compared, there is a bias for smaller sample sizes. However, for $q \geq 2$ there is a perfect agreement between the simulated and asymptotic means for all sample sizes.

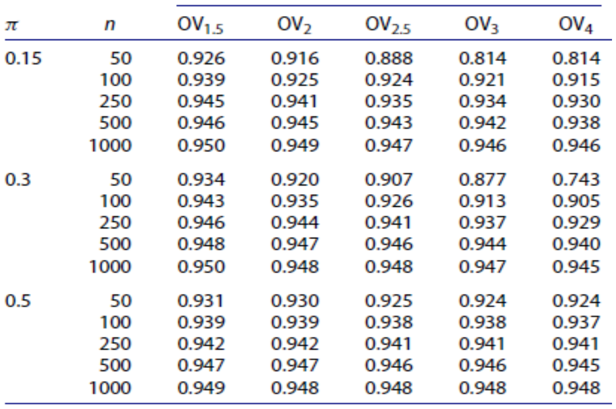
Afterwards, Weiß (2019a) computes simple plug-in CIs (level 95%) with limits $\widehat{OV}_q \pm z_{0.975} n^{-1/2} \hat{\sigma}_q$, results are given in Table 2.3. When sample sizes increase, coverages approach to 0.95. There are deviations for $n \leq 100$. While q increases, coverages become worse.

Table 2.3 Ordinal data $x_1, \ldots, x_m$ with $i.i.d.$ rank counts $I, I \sim Bin(m,\ \pi)$; asymptotic vs. simulated mean (‘a. M’ vs. ‘s. M’) of $\widehat{OV}_q$. (Source: Weiß (2019a))



		$OV_{1.5}$		OV_2		$OV_{2.5}$		OV_3		OV_4	
π	n	a. M	s. M	a. M	s. M	a. M	s. M	a. M	s. M	a. M	s. M
0.15	50		0.252	0.219	0.219	0.194	0.194	0.174	0.174	0.147	0.147
	100		0.255	0.221	0.221	0.195	0.196	0.176	0.176	0.148	0.148
	250	0.257	0.256	0.222	0.222	0.197	0.197	0.177	0.177	0.149	0.149
	500		0.256	0.223	0.223	0.197	0.197	0.177	0.177	0.149	0.149
	1000		0.257	0.223	0.223	0.197	0.197	0.177	0.177	0.149	0.149
	∞	0.257		0.223		0.197		0.177		0.149	
0.3	50		0.358	0.331	0.330	0.306	0.306	0.285	0.285	0.250	0.250
	100		0.361	0.334	0.334	0.310	0.310	0.289	0.289	0.254	0.254
	250	0.363	0.362	0.336	0.336	0.313	0.313	0.291	0.291	0.256	0.256
	500		0.363	0.337	0.337	0.313	0.313	0.292	0.292	0.256	0.256
	1000		0.363	0.337	0.337	0.314	0.314	0.292	0.292	0.257	0.257
	∞	0.363		0.338		0.314		0.293		0.257	
0.5	50		0.413	0.379	0.380	0.358	0.356	0.339	0.338	0.313	0.313
	100		0.419	0.384	0.383	0.360	0.359	0.342	0.341	0.318	0.318
	250	0.428	0.423	0.386	0.386	0.361	0.361	0.344	0.344	0.320	0.321
	500		0.425	0.387	0.387	0.362	0.362	0.344	0.344	0.321	0.321
	1000		0.426	0.387	0.387	0.362	0.362	0.345	0.344	0.322	0.322
	∞	0.428		0.388		0.362		0.345		0.322	

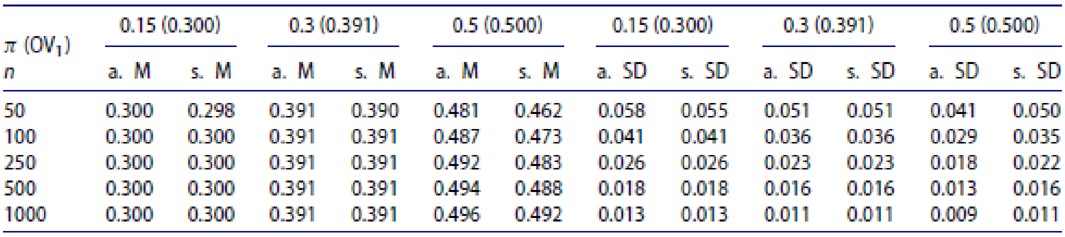
Table 2.2 Ordinal data $x_1, \ldots, x_m$ with $i.i.d.$ rank counts $I, I \sim Bin(m,\ \pi)$; coverage rates of plug-in 95% CIs based on $\widehat{OV}_q$. (Source: Weiß (2019a))



π	n	$OV_{1.5}$	OV_2	$OV_{2.5}$	OV_3	OV_4
0.15	50	0.926	0.916	0.888	0.814	0.814
	100	0.939	0.925	0.924	0.921	0.915
	250	0.945	0.941	0.935	0.934	0.930
	500	0.946	0.945	0.943	0.942	0.938
	1000	0.950	0.949	0.947	0.946	0.946
0.3	50	0.934	0.920	0.907	0.877	0.743
	100	0.943	0.935	0.926	0.913	0.905
	250	0.946	0.944	0.941	0.937	0.929
	500	0.948	0.947	0.946	0.944	0.940
	1000	0.950	0.948	0.948	0.947	0.945
0.5	50	0.931	0.930	0.925	0.924	0.924
	100	0.939	0.939	0.938	0.938	0.937
	250	0.942	0.942	0.941	0.941	0.941
	500	0.947	0.947	0.946	0.946	0.945
	1000	0.949	0.948	0.948	0.948	0.948

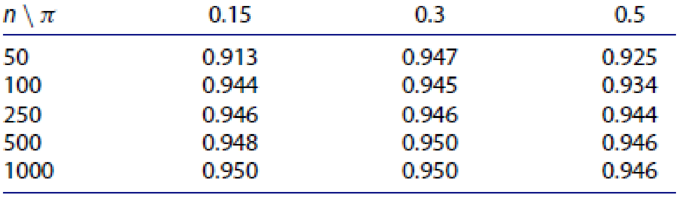
Later in the study, Weiß (2019a) analyzes the approximation quality of $\widehat{\mathrm{OV}}_1$. Table 2.4 shows the results with the means and standard deviations. Approximate standard deviation works well for $n \geq 100,\ \pi \neq\ 0.5$. When $\pi = 0.5$, deviations are observed because of the discontinuity problem. In Table 2.4, asymptotic and simulated means are compared for the $\widehat{OV}_1$ and there is a bias for smaller sample sizes, $\pi \neq\ 0.5$. For $\pi = 0.5$ bias is more notable. Weiß (2019a) indicates that extent of bias correction is not sufficient for $q = 1$. In Table 2.5, the plug-in $95\% - CI$s of $\widehat{OV}_1 \pm z_{0.975} n^{-1/2} \hat{\sigma}_1$ works well. Two exceptions are when $\pi = 0.5$ and smaller sample sizes for $\pi = 0.15$.

Table 2.4 Ordinal data $x_1, \dots, x_m$ with $i.i.d.$ rank counts $I, I \sim Bin(m,\ \pi)$; asymptotic vs. simulated mean ('a. M' vs. 's. M') and standard deviation ('a. SD' vs. 's. SD') of $\widehat{OV}_1$. (Source: Weiß (2019a))



π (OV$_1$)	0.15 (0.300)		0.3 (0.391)		0.5 (0.500)		0.15 (0.300)		0.3 (0.391)		0.5 (0.500)	
n	a. M	s. M	a. M	s. M	a. M	s. M	a. SD	s. SD	a. SD	s. SD	a. SD	s. SD
50	0.300	0.298	0.391	0.390	0.481	0.462	0.058	0.055	0.051	0.051	0.041	0.050
100	0.300	0.300	0.391	0.391	0.487	0.473	0.041	0.041	0.036	0.036	0.029	0.035
250	0.300	0.300	0.391	0.391	0.492	0.483	0.026	0.026	0.023	0.023	0.018	0.022
500	0.300	0.300	0.391	0.391	0.494	0.488	0.018	0.018	0.016	0.016	0.013	0.016
1000	0.300	0.300	0.391	0.391	0.496	0.492	0.013	0.013	0.011	0.011	0.009	0.011

Table 2.5 Ordinal data $x_1, \dots, x_m$ with $i.i.d.$ rank counts $I, I \sim Bin(m,\ \pi)$; coverage rates of plug-in 95% CIs based on $\widehat{OV}_1$. (Source: Weiß (2019a))



$n \setminus \pi$	0.15	0.3	0.5
50	0.913	0.947	0.925
100	0.944	0.945	0.934
250	0.946	0.946	0.944
500	0.948	0.950	0.946
1000	0.950	0.950	0.946

Weiß (2019a) explains that asymptotic approximations of OV_q's are more problematic for $q < 2$ because of the discontinuity and poles and suggest choosing $q \geq 2$. Also, Weiß investigates the OV_∞, and do not suggest selecting q too large because too much weight is put into sole deviations of p_0, p_m from 0.5, so selecting q from the interval $[2; 3]$ might be useful.

Since data is obtained from experimental design in this study, the data consists of small sample sizes, around $n = 50$. Based on Weiß (2019a), it is investigated which value of q would be appropriate for this study. Casella and Berger (2002) explains that a good estimator should have small bias and small variance. Based on this explanation, the quality of the estimation for the small sample size, $n = 50$, is summarized by looking at the asymptotic and simulated distributions of Weiß (2019a) and is shown in Table 2.6 below.

Table 2.6 Approximation Performances of $\widehat{OV}_q$, for $n = 50$.

	$\boldsymbol{q = 1}$	$\boldsymbol{q = 1.5}$	$\boldsymbol{q = 2}$	$\boldsymbol{q = 2.5}$	$\boldsymbol{q = 3}$	$\boldsymbol{q = 4}$
Approximate Standard Deviation	Higher deviation	Small deviation	Small deviation	Small deviation	Small deviation	Small deviati on
Approximate Means	Higher bias	Could not be obtained	Small bias	Small bias	Small bias	Small bias

According to Table 2.1 and 2.4, higher deviations are obtained for $q = 1$ compared to $q \geq 2$. Similar situation is observed in Figures 2.2 and 2.4, for $q = 1$ bias is more notable. Also, for $q = 1.5$, approximations could not be obtained. Considering the suggestion of Weiß (2019a), $q = 1$ or $q = 1.5$ should not be chosen. In Figure 2.1 and Figure 2.2, for $q \geq 2$, there are small deviation and small bias in the standard deviations and the means, respectively. This situation is parallel to Casella & Berger (2002) definition of a good estimator, but no distinctive difference is observed for $q \geq 2$. For this reason, plug-in 95% CIs estimates are considered. Considering the values other than $q = 1.5$ in Table 3, coverage deteriorates as q values increase for $n = 50$. However, $q = 2$ performs better than the others.

As noted previously, minimum and maximum dispersions for ordinal data are associated with the one-point distribution and polarized distribution, respectively (Kvålseth, 2011; Weiß, 2019a). Dispersion values obtained from different ordinal dispersion measures for different probability distributions are analyzed in Appendix A. It is observed that the $\mathrm{LOV}, \mathrm{COV}, \mathrm{LSQ}, \Delta^*, \mathrm{OV}_{2.5}, \mathrm{OV}_3, \mathrm{V[I]}, \mathrm{disp}_{0,2}$, and variance show similar behavior under different distributions. For location measures, median produced different results than the expected value and rounded$-E[I]$. Correlation between the ordinal dispersion measures is also examined in Appendix A. All dispersion measures are highly correlated with each other. For the location measures, while expected value and rounded$-E[I]$ are highly correlated, median does not have linear relationship with these measures. Following the suggestion of Weiß (2019a) and based on the analysis of ordinal dispersion measures presented in Appendix A, OV_2, a.k.a. COV, is selected as dispersion measure for the rest of the study.

Weiß (2019b) suggests a location measure for ordinal categorical data. He explains rank-count approach as, for an ordinal random variable Y with outcomes $\boldsymbol{S} = \{s_0 < \cdots < s_m\}$, rank-count I is defined by the relation $Y = s_I$ so that I counts by how many categories Y departs from the lowest category. Based on this definition, the proposed measure uses distances in range S. The block and Euclidean distance examples are given below.

$$d_{O,1}(s_i, s_j) = |i - j| \tag{2.14}$$

$$d_{O,2}(s_i, s_j) = (i - j)^2 \tag{2.15}$$

For the calculation of location for random variable Y, the expected distance is used (Weiß, 2019b).

$$\operatorname*{argmin}_{y \in S} E[d(Y, y)] \tag{2.16}$$

If the squared Euclidean distance is used, then the location value is the rounded value of $E[I]$.

$$rounded - E[I] = k - \sum_{i=0}^{k-1} f_i \tag{2.17}$$

In the same study Weiß (2019b) also suggests a dispersion measure.

$$disp_{O,2} = 2V(I) \tag{2.18}$$

$$V(I) = \sum_{i=0}^{k-1} f_i \left(1 - \sum_{j=0}^{k-1} f_j \right) + 2 \sum_{i=1}^{k-1} \sum_{j=0}^{i-1} f_j \tag{2.19}$$

The normalized value is obtained with $(4/k^2)V(I)$. The location measures, rounded- $E[I]$, median and expected value, are also compared in this master thesis (see Appendix A). The rounded- $E[I]$ and expected value have higher correlation and median is relatively less correlated with these measures. Based on this result and suggestion of Stevens (1946), median is used as location measure for the rest of the study.

2.3 Data Mining Algorithms and Ensemble Learning

Ensemble learning is a technique that aims to achieve better prediction performance by combining multiple base learners (Sagi & Rokach, 2018). Base learners (also known as weak learners) perform only slightly better than a random guess. However, a combination of these constructs a strong learner which is prediction performance is significantly better.

Base learners are constructed with machine learning algorithms such as Naive Bayes, Decision Tree, Neural Network, etc. If the same learning algorithm is used to train all core learners, it is called *homogeneous ensemble,* if different algorithms are used, it is called *heterogeneous ensemble* (Zhou, 2012).

One of the benefits of using ensemble learning is it reduces the prediction error. Prediction error consists of bias and variance, and there is a trade-off between them. Bias is a measure of how close the learning algorithm is to the target value. A high bias indicates that the model does not fit the training data well, in which case underfitting occurs. On the other hand, variance measures how different the algorithm makes predictions for training data of the same size. High variance indicates that the model is overfitting the training data, and in this case, the model cannot be generalized. Low bias and low variance are two important properties of a well-constructed model. Rokach (2019) explains that using ensemble learning reduces the generalization error of individual learners because the different types of learners have different biases, and this diversity helps to decrease variance without increasing the bias. Therefore, one of the essential properties of good ensemble learning is diversity, that is, contributing base learners make different errors for the same instance. Another important feature of a good ensemble is accuracy; an ensemble should have base learners which are accurate (Rokach, 2019; Sagi & Rokach, 2018; Zhou, 2012).

According to Sagi and Rokach (2018) and Zhou (2012) ensemble techniques are divided into two groups: dependent and independent techniques. In dependent techniques, the goal is to correct the error in the algorithm throughout the iterations. Therefore, base learners are created to correct the errors of the previous one. There is sequential learning in the dependent techniques. In the independent techniques, each base learner is created and trained independently from each other. Since there is parallel training in independent methods, the final prediction is made by majority voting or averaging.

2.3.1 Bagging

Bagging, abbreviation of bootstrap aggregating, is proposed by Breiman (1996). It is an example of independent ensemble techniques. In the bagging procedure, first, the dataset is divided into n different subsets using the bootstrap sampling method. For each subset, data is generated randomly by sampling with replacement. Therefore, some instances in the original dataset may appear multiple times in one bootstrap replication, while others may not appear at all in that replication. However, in the bootstrap process, too many subsets are generated, and it increases the probability that an instance will appear in at least one subset. In the training stage n base learners are trained using n subsets. The final decision, aggregating, is made by combining the outputs of the base learners: voting for classification, averaging for regression. Kumar and Jain (2020), explains that randomly generated subsets lead to a diverse ensemble, resulting in better performance than from a single model. An illustration of the bagging algorithm (Kumar & Jain, 2020) is shown in Figure 2.1.

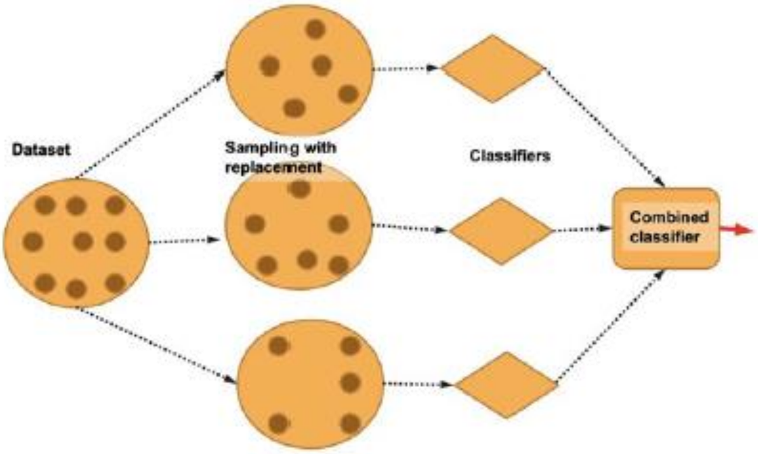


Figure 2.1 Bagging with sampling with replacement (Source: (Kumar & Jain, 2020))

2.3.2 Random Forest

Random Forest (Breiman, 2001) is an independent type of ensemble learning technique and is preferred by many people due to its simplicity and good performance. It uses decision trees as base learners. In the algorithm, random subsets are generated with the same distribution as the original dataset. Decision trees are trained with the subsets. After a large number of trees have been grown, voting is used to make the final prediction. Breiman (2001) explains that the Random Forest algorithm will not present an overfitting problem because it uses the Strong Law of Large Numbers, so that generalization error always converges.

Breiman (2001) emphasizes that Random Forest includes two types of randomness. The first one is bagging, that is, a new training subset is created by drawing instances from the original data. The training subset is generated with replacement, so some of the samples may be drawn several times while some of them may not appear in that training set at all. Then a decision tree is constructed with that subset using random feature selection, which is the second randomness factor. A random subset of features is used in each node to make an optimal split. Afterward, trees are grown to the maximum possible size and remain unpruned.

Breiman (2001) explains a useful property of the Random Forest: out-of-bag (OOB) samples. 67% of the samples in the original dataset are used in the training of a base learner, while the other 33% are not included in the training. This 33% of the samples are called OOB samples. Since the base learner does not see the OOB samples throughout the training, these samples can be used to evaluate the learner's performance without requiring an external cross-validation method. Therefore, the test set is not necessary for Random Forest. The training and test set separation can be a problem when there are few instances in the dataset. However, Random Forest solves this problem using OOB samples. The performance of an ensemble can be evaluated by averaging the OOB errors (OOB estimates) of each learner (Géron, 2019). In addition to this, Breiman (2001) mentions that cross-validation includes bias; on the other hand, OOB estimates are unbiased.

According to Breiman (2001) Random Forest has comparable accuracy with AdaBoost, it is robust to outliers and noise. In AdaBoost, misclassified instances are weighted more, and the algorithm focuses on these instances throughout the iterations. In this sense, AdaBoost is sensitive to noisy data and outliers. Since Random Forest does not use weights, the effect of noise is less noticeable. Furthermore, Random Forest works faster, it is simple to implement, and it can be parallelized (Breiman, 2001). Using random subsets and random features makes the trees less correlated and this helps to increase diversity. Also, Random Forest contains fewer parameters to be optimized. In this sense, it is easier to tune (Sagi &

Rokach, 2018). The two important hyperparameters of Random Forest are number of trees and number of attributes used in each node (Breiman, 2001; Sagi & Rokach, 2018). The Scikit-learn library provides a *"class weight"* option which is helpful when data is imbalanced. It gives weights to the classes inversely proportional to their frequencies as given in Equation (2.20).

$$w_k = \frac{number\ of\ samples}{number\ of\ classes\ \times bin\ count\ of\ class\ k} \tag{2.20}$$

where w_k is the weight of class k. Moreover, these class weights can be adjusted for bootstrap samples for every tree in the forest with the *"balanced subsample"* option (Pedregosa et al., 2011). Random Forest provides other useful properties such as feature importance, and proximity plots to deal with missing data.

2.3.3 Boosting

Boosting is simply the procedure of transforming weak learners into strong learners. It is an example of dependent ensemble techniques, so sequential training is used in boosting. At the beginning of the algorithm, a subset is generated randomly from the original dataset. A model is fitted with that subset. According to the output of the model, wrongly predicted instances are determined. A new subset is generated so that it includes misclassified instances of the first iteration and the instances which are not in the previous subset. In the second learner, a new model is built with this new subset to fix errors of the first learner. The algorithm aims to correct the mistakes of the previous learner by working in iterations. In the end, the final model is obtained which is the average of these models.

2.3.4 AdaBoost

AdaBoost (Freund & Schapire, 1997), abbreviation for adaptive boosting, is one of the most popular boosting algorithms. AdaBoost uses weights to obtain better-performed models. First, equal weights are assigned to each instance in the dataset. A weak learner is trained and according to the prediction results, misclassified instances are determined. Incorrectly predicted instances are given more weight, and similarly correctly classified instances are weighted less. This way, the weak learner concentrates more on the hard instances. Afterward, the second weak learner is trained with the new weighted dataset. This process is repeated until the specified number of iterations or until the model performance reaches the desired level. Voting is used to obtain the final prediction results (Freund & Schapire, 1999; Kumar & Jain, 2020).

In AdaBoost, not only instances are weighted, but also weak learners are weighted according to their accuracy. Higher weight is assigned to the more accurate predictor. AdaBoost provides a decrease in bias and performs well both for regression and classification. Freund and Schapire (1999) explains the advantages of AdaBoost as,

- It is fast and easy to use,
- It does not require hyperparameter tuning,
- It is flexible, you can combine any method to use as a base learner.

In Figure 2.2, a visualization of the AdaBoost algorithm (Kumar & Jain, 2020) is given.

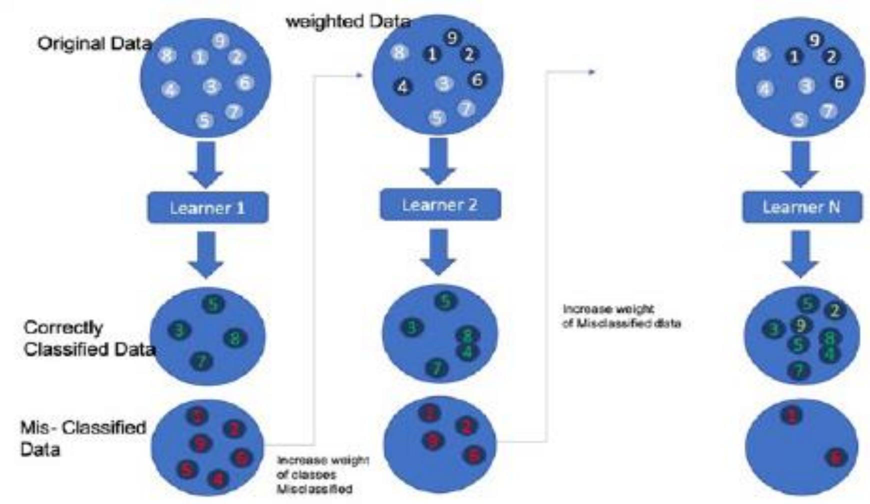

Original Data
weighted Data
Learner 1
Learner 2
Learner N
Correctly Classified Data
Mis- Classified Data
Increase weight of classes Misclassified
Increase weight of Misclassified data

Figure 2.2 AdaBoost boosting (Source: Kumar & Jain, 2020))

2.3.5 Gradient Boosting Machines

Gradient Boosting Machines (GBM) (Friedman, 2001) is a nonparametric and dependent type of ensemble technique. It typically uses decision trees as base learners. As with the other boosting methods, in GBM, each base learner is iteratively added to correct the mistakes of the previous learner. However, the difference from the AdaBoost is that instead of building a model by weighting the incorrectly predicted observations, a model is built on the residuals of the previous model.

Gradient descent optimization algorithm is used in GBM. The objective is to minimize the loss function when adding a new tree to the ensemble. This loss function is arbitrary and differentiable. In the final prediction, outputs of the base learners are aggregated in an additive manner (Rokach, 2019).

In machine learning, the loss function is a metric that shows how well the model fits each sample. It measures the deviation between the predicted output and the actual output. A loss function can be described simply as

$$L(y, \hat{y}) \tag{2.21}$$

Where y is the actual target value and $\hat{y}$ is the predicted value of the y. There are many loss functions available. The most popular of these are the squared-error (aka L_2-loss) $(y - \hat{y})^2$ and absolute error (aka L_1-loss) $|y - \hat{y}|$ for regression and the negative binomial log-likelihood loss function for binary classification problems (Friedman, 2001). The loss should be small as it expresses the deviation from the true value.

Rokach (2019) explains the difference between GBM and other ensemble techniques as GBM uses many simple models, while other ensemble techniques use fewer but more complex models. Therefore, choosing the right number of predictors is an important parameter for GBM (Friedman, 2001; Rokach, 2019). Using large number of predictors can cause overfitting while using a small number of predictors can cause underfitting.

Friedman (2001) lists the advantages of using gradient boosting of regression trees:

- Robustness: GBM gives the same result in any transformation of the input variable (feature), it does not require nonlinear transformation, and it is strong against skewed distributions.
- Internal feature selection: GBM has internal feature selection methods, it is robust to the addition of unimportant features. It does not require external feature selection.
- Handling missing values: It is also good at handling missing values; it can fill in missing observations without the need for an extra imputation method.
- Accuracy and stability: Inaccuracy and instability are two problems that the decision trees are suffering from. However, with the GBM, the effect of these problems is mitigated. Friedman (2001) also mentions one disadvantage of the tree boosting system: single decision trees can be easily interpreted while boosted trees lack interpretability.

Since it is a boosting type of technique GBM can overfit. To overcome the overfitting problem, using regularization provides useful results. Regularization works in a way that reduces model complexity, which is defined as a function of the number of leaves in the trees in GBM. By adding a penalty, instead of just minimizing the loss function in the objective function, both the loss function and complexity can be minimized (Rokach 2019). This is the working principle of XGBoost.

2.3.6 XGBoost

Chen and Guestrin (2016) introduces a new scalable boosting system called eXtreme Gradient Boosting (XGBoost). It is a tree-based ensemble learning algorithm based on gradient boosting. Similar to GBM, XGBoost minimizes the loss function in the objective. However, it also tries to minimize the penalty in the objective function. The penalty is a function that is used to prevent the problem of overfitting and makes the model more conservative. With both algorithmic and system improvements, Chen and Guestrin (2016) makes XGBoost much faster and more accurate than many existing machine learning algorithms. These improvements are

- Regularized objective function to prevent overfitting,
- Approximate greedy algorithm to increase the speed,
- Sparsity-aware Split Finding algorithm to handle missing data,
- Weighted Quantile Sketch to help approximate greedy algorithm,
- Parallel Learning to make computation faster,
- Cache-aware Access,
- Blocks for Out-of-core computation.

The first step in XGBoost is to make an initial prediction. The initial prediction is 0.5 by default, it does not matter what number the initial prediction is as it will converge to the correct result. Then, residuals are calculated by subtracting the initial prediction from the actual target values. In the next step, a decision tree is started to be built. All the residuals are added to the root node forming the tree, and the quality score (also known as the similarity score) is calculated for the root node. Then, a split is selected among the candidate splits using the Approximate Algorithm: for each of the candidate splits, the residuals in the first stage are placed on the child nodes and the quality score is calculated for these nodes separately. The gain calculation for each candidate split is obtained by subtracting the sum of the quality scores of the right and left nodes from the root node. The candidate with the highest gain is selected for splitting, and decision trees are constructed this way. Each tree is grown to compensate for the mistakes of its predecessor. The final decision is obtained by summing the predictions on the respective leaves of the trees. Figure 2.3 shows how the final decision is made (Chen & Guestrin, 2016).

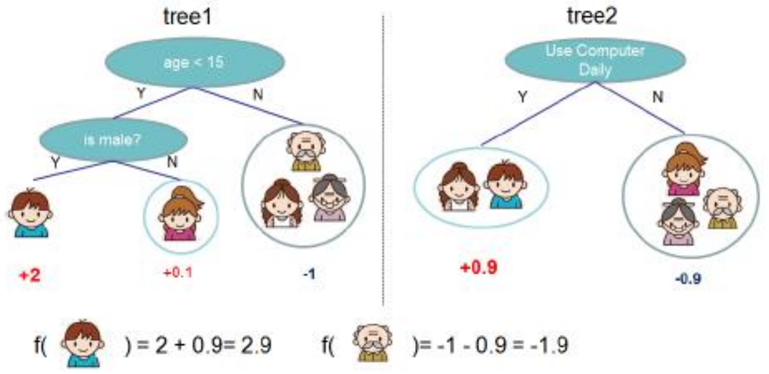


Figure 2.3 Tree Ensemble Model. The final prediction for a given example is the sum of predictions from each tree. (Source: (Chen & Guestrin, 2016))

In an algorithmic framework, XGBoost is the optimized and enhanced version of the GBM algorithm. The most obvious difference is that it uses a regularized objective function which Chen and Guestrin (2016) defines as follows:

$$L(\phi) = \sum_{i} l(y_i, \hat{y}_i) + \sum_{k} \Omega(f_k) \qquad (2.22)$$

$l(y_i, \hat{y}_i)$ is the differentiable convex loss function which measures how close predicted value of the target, $\hat{y}_i$, is to the actual value of target, y_i. $\Omega(f_k)$ in Equation (2.22) is a regularization term, which is used to prevent overfitting. It is defined as:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2 \qquad (2.23)$$

T is the number of leaves in the tree, γ and λ are shrinkage parameters and w is the weight of the leaves. The objective function in Equation (2.22), without the $\Omega(f_k)$ term same is as the GBM's objective function.

Since XGBoost is a boosting-type ensemble technique, it trains decision trees in an additive manner. So, in t-th iteration objective function is:

$$L^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \qquad (2.24)$$

$\hat{y}_i{}^{(t-1)}$ is the predicted value of i-th observation at the previous iteration. The goal is adding f_t so that the minimizes the objective function. This also means that the function that will most improve the model is added in a greedy way. To solve this formula, Second-order Approximation is used because it will take a long time to calculate this function directly. In this case, the objective function becomes this,

$$L^{(t)} \cong \sum_{i=1}^{n} [l(y_i, \hat{y}_i{}^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t{}^2(x_i)] + \Omega(f_t) \tag{2.25}$$

In this function, g_i and h_i are the first and second-order gradient statistics of the loss function respectively, and they are calculated as follows:

$$g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}_i{}^{(t-1)}) \tag{2.26}$$

$$h_i = \partial^2{}_{\hat{y}^{(t-1)}} l(y_i, \hat{y}_i{}^{(t-1)}) \tag{2.27}$$

In equation 2.25 $l(y_i, \hat{y}_i{}^{(t-1)})$ term becomes constant. Also, if $\Omega(f_t)$ is written as in Equation (2.23), $L^{(t)}$ becomes:

$$L^{(t)} = \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t{}^2(x_i)] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j{}^2 \tag{2.28}$$

Or simply,

$$L^{(t)} = \sum_{j=1}^{T} [(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i + \lambda) w_j^2] + \gamma T \tag{2.29}$$

Where $I_j = \{i | q(x_i) = j\}$ is the instances on the leaf j and $q(x_i)$ is a mapping function that assigns every instance, x_i, to the leaves. After solving this function, the optimal weight of a leaf (also known as output value of a leaf) is calculated as:

$$w_j{}^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \tag{2.30}$$

If ${w_j}^*$ is put in the places corresponding to w_j in the Equation (2.30) the loss function becomes,

$$L^{(t)}(q) = -\frac{1}{2}\sum_{j=1}^{T}\frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_j} h_i+\lambda} + \gamma T \tag{2.31}$$

Equation (2.31), the quality score or the similarity score is used to evaluate the quality of nodes (Chen & Guestrin, 2016). λ is a shrinkage parameter so, the higher value of λ, smaller output values for the leaves.

When constructing decision trees, a split is chosen according to gain. The gain formula is given in Equation (2.32).

$$Gain = \left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in \mathrm{I}} g_i)^2}{\sum_{i\in \mathrm{I}} h_i + \lambda}\right] \tag{2.32}$$

$$L_{split} = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in \mathrm{I}} g_i)^2}{\sum_{i\in \mathrm{I}} h_i + \lambda}\right] - \gamma \tag{2.33}$$

Where I_L and I_R are the instance sets of the left and right nodes, and $I_L \cup I_R = I$. Candidate splits are evaluated with the gain formula. The quality score of the previous node is subtracted from the quality scores of the right and left nodes. The one with the highest gain is selected for the split. Equation (2.33) is used in the pruning stage. Pruning decision is made by subtracting γ from the gain so that nodes with the negative result are discarded from the tree. Here, the γ parameter is used to provide regularization and prevent overfitting. So, the larger values of γ and λ result in shallow trees (Bentéjac et al., 2021).

2.3.6.1 Approximate Algorithm and Weighted Quantile Sketch

In GBM, every possible candidate split is evaluated and the candidate which provides the highest gain is selected for splitting. In this case, GBM uses the Basic

Exact Greedy Algorithm. According to Chen and Guestrin (2016) the problem with the Exact Greedy Algorithm is it calculates and sorts all possible candidate splits for all features. Therefore, it takes time to calculate the gain for all the candidates, short and select the one that provides maximum gain. Time complexity is quite high especially for large and high dimensional datasets or when the data does not fit entirely into memory. Thus, Chen and Guestrin (2016) proposes Approximate Algorithm to reduce the time complexity and make the calculations in a more efficient way. Instead of checking every possible candidate for splitting, the algorithm divides data into quantiles, then checks for the gain of these quantiles and selects the best one. So, the Approximate Algorithm does not guarantee the best split in the long term, however, it reduces the computation time.

To find candidate splitting points with the Approximate Algorithm, Chen and Guestrin (2016) introduces Weighted Quantile Sketch which is used with Parallel Learning. First, the dataset is divided into smaller subsets and distributed to several computers. In each computer, values are aggregated to make an approximate histogram which will be used for calculating the quantiles. In the Quantile Sketch algorithm, each quantile has an equal number of observations, however, in the Weighted Quantile Sketch, weights are assigned to the observations, and the sum of weights in each quantile equals or is close to each other. The weight of each observation equals to h_i. As the weight of an observation increases, the number of observations of the same quantile decreases, and the observation becomes much more important for splitting. Also, as the number of quantiles is increased, there will be fewer instances in a quantile and the algorithm becomes more accurate. The authors also mentioned that this approach has provable theoretical guarantees, and it maintains a certain accuracy level.

2.3.6.2 Tree Pruning Strategy

One of the differences between XGBoost and GBM is their pruning strategy. Both algorithms use Equation (2.33) to make a pruning decision. GBM uses this equation without γ parameter. Nodes with a negative score are not allowed in either algorithm. If a negative score is encountered in a node while building a tree in GBM, that node is pruned, so GBM is a greedy algorithm. On the other hand, in XGBoost the tree is grown to the predetermined maximum depth and no pruning is done at this stage. After that, scores are calculated retrospectively and if a negative score is encountered, the related node is pruned. XGBoost constructs trees using the depth-first strategy. This strategy increases the model complexity and helps to increase the model performance. Each tree in the ensemble will have a different depth level after the pruning and this leads to increased diversity.

2.3.6.3 Sparsity-aware Split Finding

Most of the real-world datasets suffer from sparsity or missing values. There are some external techniques to deal with missing data such as simple imputation, complete case analysis, using a constant to fill missing values, and multiple imputation. However, XGBoost provides a useful method to deal with missing data, Sparsity-aware Split Finding. In this way, the need for an external technique is eliminated. This method provides guidance on how to grow a decision tree with missing data, and how to fill those values.

In the Sparsity-Aware Split Finding, a tree is built, and residuals are calculated for the entire dataset. Then, missing observations are removed from the original dataset and a separate dataset is created with them. For each candidate split, the residuals of the missing data are put in the left node and right node, respectively, and gain is calculated for both. This way, the algorithm learns the best direction to impute the missing data (Chen & Guestrin, 2016). This process is repeated until all quantiles are tried and the one that provides maximum gain is selected for splitting.

2.3.6.4 Column Block for Parallel Learning

XGBoost provides not only performance improvements, but also it provides system optimization and parallelization. Generally, real-world datasets are huge, and it cannot be possible to fit all into computer memory at one time. To make calculations quickly, XGBoost uses Parallel Learning which is splitting dataset into smaller subsets and putting each subset on different computers on a network. Chen and Guestrin (2016) proposes to store data in blocks, also known as in-memory units. They explained that using blocks helps with the Approximate Algorithm because in the Exact Greedy Algorithm all data is stored in one block while in the Approximate Algorithm data can be stored in multiple blocks. Compressed column format (CSC) is being used to reduce the data sorting process and data is divided into blocks using this method. When Approximate Algorithms run, data can be split into multiple blocks to distribute among different machines to be worked on. According to Chen and Guestrin (2016) using blocks also helps column subsampling while adding randomization to the algorithm.

2.3.6.5 Casche-aware Access

A computer has a cache inside the Central Processing Unit (CPU), and the CPU uses this cache faster than any other memory in the computer. Due to XGBoost's execution speed concerns, calculations about the loss function, gains, and first and second-order gradients are done in the cache.

This method is proposed for the cache-miss occasion as a solution when data is too large for CPU cache to fit. Cache awareness is done by loading a set of data into each thread when using a greedy algorithm. For parallelized situations in approximate algorithms, the block size of the fetched data into the thread buffer plays a larger role. When it is set too low, parallelization lacks performance; on the other hand, when loaded block size is set too large, cache misses occur since the block does not fit the CPU cache. A balanced size from the test runs demonstrates the best performance among all (Chen & Guestrin, 2016).

2.3.6.6 Blocks for Out-of-core Computation

One of the main reasons for XGBoost's good performance is that it is designed to use the computer’s resources optimally. After splitting data into blocks, it stores each block on disk. For out-of-core computation, block data that is not fitted to the memory space is stored into the hard-disk. To overcome the read/write IO (input/output) operation overhead in computation time, one thread is dedicated to fetch and store data concurrently with the computational threads. *Block Compression* is used to reduce the size to speed up the reading/writing times via calculating an offset on rows and storing the indices in a 16-bit integer. *Block Sharing* is another technique that works on multiple disks storing divided data. In this method, a thread is assigned to each disk to fetch data to the shared memory space.

Besides all the algorithmic and system optimizations, XGBoost can also use the column subsampling which is taken from Random Forest. In this way, samples from the original dataset are randomly drawn without replacement, and diversity in the ensemble is increased.

2.3.6.7 Hyperparameters of XGBoost

One of the advantages of XGBoost is flexibility. It has many hyperparameters to tune and it allows users to train the model as desired. There are parameters to control the complexity of the model or to add randomness to the algorithm (XGBoost

Developers, 2020). The hyperparameters of XGBoost and the reasons for their use are explained below.

- Objective: This parameter is for what purpose the user is using the algorithm. Typically, for regression it can be set as *reg:squarederror*, for binary classification *binary:logistic*, for the multiclass classification it is *multi:softmax* can be used. *multi:softprob* can be used if the user wants to output as the probability of each data point belonging to each class, it gives $ndata * nclass$ vector as result.
- Number of estimators: It is used for deciding the number of trees in an ensemble or it can be seen as a number of boosting iterations.
- Learning Rate: (also known as *eta*): In XGBoost decision trees tend to learn quickly and it leads to overfitting. The learning rate parameter scales the trees. It is used to slow down the learning process and prevent overfitting.
- Gamma: It is a shrinkage parameter that is used in tree pruning. In the gain calculation and tree pruning stages, *gamma* is used, and nodes with the negative gains are pruned. Gamma makes algorithms robust to overfitting.
- Maximum depth of a tree: It is the depth of a decision tree and affects the model complexity. The more the maximum depth, the deeper trees are built, and models become more complex. Maximum depth parameter allows the model to better examine the relationship of features with each other.
- Minimum child weight: Basically, it is the minimum sum of weights of instances needed in a child node. It used to control overfitting.
- Subsample: Determines the subsampling ratio of the training instances. In each iteration, it is used to sample the original dataset randomly, without replacement. This way, each tree is trained with different instances, and it increases the variation in the ensemble.
- Column subsampling: It increases the diversity by adding more randomness to the models. This parameter represents column subsampling according to:
  - By tree: column subsampling ratio used to build each tree.

- By level: column subsampling ratio used in each depth of a tree.
- By node: column subsampling ratio used in each node. Random forest subsamples the features this way.

- Evaluation metric: It is used for the evaluation of test data. For regression, it uses Root Mean Square Error (RMSE), for classification negative log-likelihood, Area Under Curve (AUC), or classification error can be used.

Besides all these parameters, XGBoost allows users to prevent overfitting with *early stopping* during the training stage. Early stopping is a way to automatically find the number of estimators. It stops iterating if the validation score is not improved for a predetermined number of iterations. It prevents both overfitting and reduces training time (XGBoost Developers, 2020).

2.3.6.8 Advantages and Disadvantages of XGBoost

XGBoost has many advantages both increasing the performance of the algorithm and providing fast training. The advantages of the algorithm are listed below.

- XGBoost is a non-parametric method like other tree-based ensemble learning methods; the underlying function is not based on an assumption. Therefore, it is not necessary to scale, transform or normalize the dataset.
- XGBoost provides several parameters to control overfitting. Maximum depth of tree, learning rate, and controlling weights of child nodes are used to make the algorithm more conservative.
- XGBoost can deal with sparsity with its built-in Sparsity-aware Split Finding algorithm. It not only handles missing data or sparseness but also guides to build trees when there is missing data.
- XGBoost uses Parallel Learning so that it distributes subsets of data into several machines; multiple CPU cores are used in the training. It increases the computation speed.

- Using the Approximate Algorithm with Weighted Quantile Sketch and Parallel Learning, XGBoost finds splits more efficiently.
- XGBoost allows trees to grow to a predetermined maximum depth level. Then, it starts pruning the tree backward, calculates the gain, and removes the nodes with negative gain. Thus, the relationship of the features with each other is better examined. In addition to this, each tree has a different complexity, and it leads to increased diversity in the ensemble.
- XGBoost allows both feature and data subsampling. Each tree is trained with different observations and features which helps increase diversity and accuracy.
- XGBoost performs well not only with the large datasets but also with the small and medium datasets. Bentéjac et al. (2021), make experiments with datasets which have different sizes including relatively small ones. The small datasets contain nearly $74 - 583$ instance and $4 - 60$ features. According to the experiment's results, XGBoost performs well with the small datasets.
- XGBoost can provide good results with imbalanced data. It has a parameter to increase the weight of the minority class *scale_pos_weight* for classification.
- XGBoost uses computer resources efficiently with the Casche-aware Access, Blocks for Out-of-core Computation, and Parallel Learning.
- XGBoost has a built-in function to obtain feature importance. It can be used as a metric to evaluate a feature. The feature importance can guide other algorithms in the feature selection stage.

Besides all these advantages, XGBoost also has some disadvantages as well.

- Since it is based on a boosting procedure, XGBoost is sensitive to outliers (Rokach, 2010). In each iteration, a model is built based on the previous model's residuals. An outlier has a much larger residual than the other observations. XGBoost focuses more on outliers than the other observations. This can also cause overfitting if the XGBoost is not well-tuned (Bentéjac et al., 2021).
- XGBoost cannot handle categorical features. These types of features need to be manually converted to numeric values by the user (Haithm et al., 2021).

2.3.7 Comparison of XGBoost and Random Forest

As both XGBoost and Random Forest provide good predictive performance, one might wonder which algorithm is better in which subjects. For this reason, the comparison of the two algorithms is presented in Table 2.7.

Table 2.7 Comparison of XGBoost and Random Forest

Properties	**Random Forest**	**XGBoost**	**References**
Hyperparameter Tuning	Less hyperparameter to tune.	More hyperparameters should be optimized.	(Bentéjac et al., 2021; Rokach, 2019)
Overfitting	Harder to overfit.	May overfit.	(Rokach, 2019)
Model Complexity	A small number of deep trees to reach the best performance.	Many shallow decision trees to reach the best performance.	(Bentéjac et al., 2021; Rokach, 2019)
Categorical features	Require manual encoding.	Require manual encoding.	(Haithm et al., 2021)

Table 2.7 (cont'd) Comparison of XGBoost and Random Forest

Handling missing values	Capable to deal with.	Capable to deal with.	(Chen & Guestrin, 2016),(Tang & Ishwaran, 2017)
Computation Speed	Fast.	Faster.	(Bentéjac et al., 2021; Sagi & Rokach, 2018)
Predictive Performance	Perform well.	Perform better than Random Forest.	(Bentéjac et al., 2021; Rokach, 2019)
Distributional Assumptions	Nonparametric.	Nonparametric.	(Tattar, 2018)
Handling imbalance Data	Can deal with it.	Can deal with it.	(Bentéjac et al., 2021; Meng et al., 2018)
Working with small data	Achieve good results.	Achieve good results.	(Bentéjac et al., 2021)
Pruning Strategy	No prune.	Depth-first strategy.	(Breiman, 2001; Chen & Guestrin, 2016)
Robustness to outliers	Robust.	Sensitive.	(Bentéjac et al., 2021; Meng et al., 2018; Rokach, 2010)

According to Rokach (2019) one of the most important hyperparameters of Random Forest is the number of features to split at each node and the number of trees. However, XGBoost requires much more hyperparameters which should be carefully tuned. Bentéjac et al. (2021) compares XGBoost, Random Forest, Gradient Boosting, CatBoost, and LightBoost using 28 datasets of different sizes. Bentéjac et al. (2021) emphasizes that Random Forest with default hyperparameters can achieve comparable results with the well-tuned versions of other ensemble techniques. However, they also explain that this can be seen as a drawback because it is hard to improve the performance of Random Forest with hyperparameter optimization. Bentéjac et al. (2021) shows that well-optimized XGBoost performs better than others, while XGBoost's default hyperparameter settings perform worse than other ensemble techniques. There is a statistically significant difference between these two versions. However, when comparing the default and optimized Random Forest, it is observed that the performance decreased by 0.5% with optimized hyperparameters in more than half of the datasets used.

Moreover, Rokach (2019) explains that Random Forest is more robust to overfitting whereas boosting-type algorithms can overfit because of the dependent nature of boosting. In Random Forest, deep trees are required to reduce bias, while few trees are required to reduce variance. Conversely, XGBoost requires shallow trees to reduce variance and uses the large number of trees to reduce bias (Rokach, 2019).

Haithm et al. (2021) mentions that both XGBoost and Random Forest are not capable of working with categorical features, so categorical values should be converted to numerical values by the user. XGBoost has a Sparsity-aware Split Finding algorithm to deal with missing data. Random Forest offers algorithms for imputing missing values such as Strawman imputation, Proximity imputation, On-the-fly-imputation (Tang & Ishwaran, 2017).

XGBoost is specifically optimized for higher speed. It uses all resources of the computer efficiently and this makes XGBoost faster. Sagi and Rokach (2018) mentions that since it supports distributed systems, XGBoost is faster than the other methods. Bentéjac et al. (2021) shows that well-optimized XGBoost prediction performance is best among all ensemble techniques including Random Forest.

Since they are tree-based methods, they are both nonparametric methods (Tattar, 2018). Bentéjac et al. (2021) also compares the ability of these algorithms to deal with imbalanced data. They uses Area Under Curve (AUC) as the evaluation metric. When comparing XGBoost and Random Forest, they observed that XGBoost performed better. Bentéjac et al. (2021) compares the execution speed of these ensembles and according to their experiments, XGBoost can be trained 3.5 times faster than Random Forest. Interestingly, for multi-class classification problems, XGBoost's speed may be worse than Random Forest. The authors explain this as, in gradient-boosting type algorithms, one base learner is trained per class. Meng et al. (2018) conducts a study to predict whether a traffic accident will occur on the highway within a one-hour period. They treat this study as a binary classification problem. Since the number of samples of the negative class in the dataset is more than the number of samples of the positive class, the problem of imbalanced data arises. To solve this, Meng et al. (2018) uses Random Forest, Gradient Boosting, and XGBoost algorithms. They give weight to the samples of positive and negative class, (higher weight assigned to the positive class) using $scale\_pos\_weight$ parameter. According to Meng et al. (2018) weighting samples yield significantly better results than random undersampling and oversampling. As a result of their study, while GBM gives the best performance among all, XGBoost outperforms Random Forest.

Bentéjac et al. (2021) makes experiments with many datasets, including small datasets. According to the results of their study, both Random Forest and XGBoost performed well on small datasets. Although in most of the datasets, XGBoost outperforms Random Forest, the prediction results are close to each other.

In Random Forest, trees are grown to the maximum possible size, and pruning is not performed (Breiman, 2001). On the other hand, trees start to be pruned after reaching the fixed maximum depth in XGBoost (Chen & Guestrin, 2016). Random Forest is robust to outliers due to its randomness (Breiman, 2001; Meng et al., 2018), whereas XGBoost is sensitive to outliers because of its dependent tree structure (Rokach 2010).

2.3.8 Comparison of Logistic Regression and Random Forest

Logistic Regression is a parametric method which is used when the dependent variable is categorical (either binary, ordinal or categorical). The dependent variable in an Ordinal Logistic Regression model is the probability that the observed category for the response, Y, is less than or equal to a certain value, and the model associates this probability with a linear function of the controllable factors through a link function, $f(.)$, as shown in Equation (2.34) (Hosmer and Lemeshow 2000).

$$f\left(\hat{P}(Y(\boldsymbol{x}) \leq i)\right) = \hat{\gamma}_i + \widehat{\boldsymbol{\beta}}\boldsymbol{x}, \;\; i = 1,2,\dots,m \tag{2.34}$$

where γ_i denotes the intercept for the model for category i, and $\boldsymbol{\beta}$ represents the vector of model coefficients. The coefficients, $\boldsymbol{\beta}$, are estimated with the Maximum Likelihood Estimation (MLE). Finally, the parameter significance test is carried out to determine whether the independent variables have a significant effect on the response (Dwidarma et al., 2021). There are different link functions available such as: logit, normit (probit), gompit (complementary log log) and so on. The selection of link functions is summarized by McCormick et al. (2017) and Orme and Combs-Orme (2009) as given in Table 2.8. Since the data for the categorical RPD problems mostly involve the cases where the higher categories are seen more frequently in the data than the others, using complementary log-log would be more appropriate.

Table 2.8 Link Functions for Ordinal Logistic Regression (Source: (McCormick et al., 2017))

Function	**Form**	**Typical Application**
Logit	$f(y) = \ln\left(\frac{y}{1-y}\right)$	Evenly distributed categories
Complementary log-log	$f(y) = \ln(-ln(1-y))$	Higher categories more probable
Negative log-log	$f(y) = -\ln(-\ln(y))$	Lower categories more probable
Probit	$f(y) = \phi^{-1}(y), where\ \phi^{-1}$ is the inverse standard normal cumulative distribution function.	Latent variable is normally distributed.
Cauchit (inverse Cauchy)	$f(y) = \tan(\pi(y-0.5))$	Latent variable has many extreme values.

If the complementary log-log link is used, the probability that the observed category is less than or equal to a certain value at $\boldsymbol{x}$ is obtained by Equation (2.35).

$$\hat{P}(Y(\boldsymbol{x}) \leq i) = 1 - exp\left(-\ exp\left(\hat{\gamma}_i + \sum_{i=1}^{p} \hat{\beta}_i x_i\right)\right) \tag{2.35}$$

Logistic Regression holds assumptions that there is a linear relationship between independent variables and the link function of the response, there are no extreme outliers, there is no multicollinearity between independent variables and the sample size is sufficiently large. These are the limitations of Logistic Regression. On the other hand, Random Forest and XGBoost are the nonparametric methods which do not depend on any assumptions.

In the literature, Logistic Regression and Random Forest are compared by many researchers. If the number of observations is less than the number of independent variables, Logistic Regression should not be used (Yoo et al., 2012). Sometimes this can be the case for RPD studies because experimental data may be limited, and the controllable factors (independent variables) can be larger than the number of experimental points. Gülbudak Dil (2018) compares performance of Random Forest and Logistic Regression. She mentioned that Random Forest generally produced better results than Logistic Regression, for three different experimental datasets.

Logistic regression groups missing values in a separate class and labels them as "unknown". Random Forest handles missing values automatically. Variable selection with Random Forest is easier than Logistic Regression (Geng, 2006). Logistic Regression uses a stepwise option for this purpose, but this option considers only a significance level for entering attribute, as a result it may select a noise variable which decreases the predictive power. Also, as mentioned in the above, Random Forest does not face an overfitting problem, whereas Logistic Regression may overfit if the dataset contains a large number of variables (Geng, 2006).

The Scikit-learn library proposes a *"class weight"* option for Logistic Regression to handle the imbalance data problem by giving weight to classes. As it is in the Random Forest, class weights are inversely proportional to the frequency of the classes (Pedregosa et al., 2011). As a result, both algorithms have many advantages and may have better features over the other.

2.3.9 Classification Performance Metrics

For all the datasets, there are more than two classes in this study. Therefore, all the case studies involve multiclass classification. In addition to that, there are imbalanced data problems, as well. Choosing the right evaluation metrics plays an important role in this case. For this purpose, imbalanced data metrics are searched. He and Ma (2013) divides metrics into two categories: threshold metrics and ranking metrics. Threshold metrics are accuracy, error rate, Cohen's kappa, sensitivity, specificity, precision, recall, geometric mean (G-mean) and F-measure. The most popular ranking metrics are Receiver Operator Characteristic Curve (ROC) and Area

Under ROC Curve (AUC). He and Ma, (2013) mentioned that most of the metrics are highly correlated where the data is balanced. However, these correlations are weakened under an imbalanced data situation. The threshold metrics are derived from the confusion matrix which is the table that includes actual and predicted class values. It includes True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). True Positives and True Negatives are correct classifications. False Positives are the number of negative instances which are incorrectly classified as positive. False Negatives are the number of positive instances which are incorrectly classified as negative. For n-class classification problem, confusion matrix is obtained as an $n \times n$ matrix which is explained in Table 2.9 (He & Ma, 2013).

Table 2.9 Confusion matrix of n −class Classification

		Predicted				
		Class$-\boldsymbol{A}$	**Class**$-\boldsymbol{B}$	$\cdots$	**Class**$-\boldsymbol{n}$	**Total**
Actual	**Class**$-\boldsymbol{A}$	N_{11}	N_{12}	$\cdots$	N_{1n}	C_A
	Class$-\boldsymbol{B}$	N_{21}	N_{22}	$\cdots$	N_{21}	C_B
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
	Class$-\boldsymbol{n}$	N_{n1}	N_{n2}	$\cdots$	N_{nn}	C_n
	Total	C_A	C_B	$\cdots$	C_n	

Positive class and negative class are the terms used for binary classification. In the multiclass case, the class of interest can be seen as positive class while others negative class. The diagonal of the matrix shows the number of correctly predicted observations, also called as True Positive (TP), for each class, such as N_{11} and N_{22}. Other elements of the matrix represent the number of incorrectly classified observations. True Negative (TN) is the correctly classified negative class observations by the classifier. False Positive (FP) is an outcome where the model incorrectly predicts the positive class. In Table 2.4, N_{12}, N_{1n}, are the FP outcomes for Class$-A$. Another term derived from the confusion matrix is False Negative (FN). FN is an outcome where the model incorrectly predicts the negative class. N_{21} and N_{n1} are incorrectly classified as class$-A$, in this case they can be labeled as FN. Summation of the rows or columns for each class is equal to each other and shows the number of total observations for that class, such as C_A, C_B.

Selection of the evaluation metric is an important issue for the imbalanced data. For the classification algorithms, one of the most preferred evaluation metrics is the accuracy score. However, it may provide unrealistic results when working with imbalanced data. Since most of the instances belong to the majority class, the accuracy score will be high, but this is not reflecting the real case (Brodersen et al., 2010; He & Ma, 2013; Hossin & Sulaiman, 2015). Accuracy score is formulated as:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.36}$$

Precision is the rate of TP among all the positive labeled instances. It shows what percentage of samples that the classifier labels as positive is actually positive. Recall, also called sensitivity or True Positive Rate (TPR), shows the ratio of how many of the positive instances are labeled as actually positive. Specificity is the negative class version of recall. It is the ratio of correctly classified negative class observations. They are formulated as follows (He & Ma, 2013).

$$Precision = \frac{TP}{TP + FP} \tag{2.37}$$

$$Recall/Sensitivity/TPR = \frac{TP}{TP + FN} \tag{2.38}$$

$$Specificity\ /TNR\ = \frac{TN}{TN + FP} \tag{2.39}$$

G-Mean is another evaluation metric based on TPR and True Negative Rate (TNR) and it considers the performance of both minority and majority classes. It is a geometric mean of sensitivity and specificity. Since it gives equal importance to all classes, g-mean is more sensitive metric than accuracy (He & Ma, 2013).

$$G - Mean = \sqrt{TPR\ \times TNR} \tag{2.40}$$

F-measure is basically the harmonic mean of precision and recall. There is a trade-off between these two metrics. If the precision is high, then the recall tends to be low (Meng et al., 2018). In the F-measure calculation β is the weighting parameter and can take values such as $0.5, 1, 2$. He and Ma (2013), defines f-measure as more advanced version of g-mean because of the weighting parameter. F-measure is calculated as:

$$F - measure = \frac{(1 + \beta^2) \times Precision \times Recall}{(\beta^2 \times Precision) + Recall} \tag{2.41}$$

The Cohen's Kappa, expresses the level of agreement between two raters. Cohen's kappa measures how much better the classifier performance than the random guess, according to the frequency of each class. This is why it is preferred in imbalanced data situations. It takes values between -1 to 1. Cohen's Kappa is calculated as (Fernández et al., 2018; Rabby et al., 2020):

$$K = \frac{P_0 - P_e}{1 - P_e} \tag{2.42}$$

P_0 is the observed agreement and P_e is the expected agreement.

$$P_0 = \frac{TP + TN}{N} \tag{2.43}$$

$$P_e = \frac{(TP + FN)}{N} \times \frac{(TP + FP)}{N} + \frac{(FP + TN)}{N} \times \frac{(FN + TN)}{N} \tag{2.44}$$

where N is the total number of observations. According to Landis and Koch (1977), it can be categorized as:

Table 2.10 Categorization of Kappa Score

Kappa Score	Agreement Level
0.8-1.0	Almost perfect
0.6-0.8	Substantial
0.4-0.6	Moderate
0.2-0.4	Fair
0.0-0.2	Slight
<0.0	Poor

Since accuracy can be misleading in case of imbalanced data classification evaluation, another alternative is balanced accuracy. It reduces the bias of the accuracy score, basically it is the average of TP and TN's (Kidando et al., 2021). For n −class classification is formulated as follows (Brodersen et al., 2010; He & Ma, 2013):

$$\frac{1}{n}(\frac{N_{11}}{C_A}+\frac{N_{22}}{C_B}+\cdots+\frac{N_{nn}}{C_n}) \tag{2.45}$$

He and Ma (2013) criticizes threshold metric because they assume that class imbalance ratio in the training data matches with the test data, however this may not always be the case.

ROC is used for visual comparison of classification models. It shows the performance of the models in a coordinate system and a classifier which is close to the upper left corner has better performance. It uses False Positive Rate (FPR) and TPR in the x and y axis in the coordinate system, respectively. The most satisfying results are obtained when classifier's performance is near to $(0,1)$. ROC basically shows the trade-off between TPR and FPR. According to He and Ma (2013), ROC is well suited to imbalance data because it evaluates performance of each class with two different measures which is where it differs from other metrics discussed above. The calculation of FPR is as follows:

$$FPR = \frac{FP}{TN + FP} \tag{2.46}$$

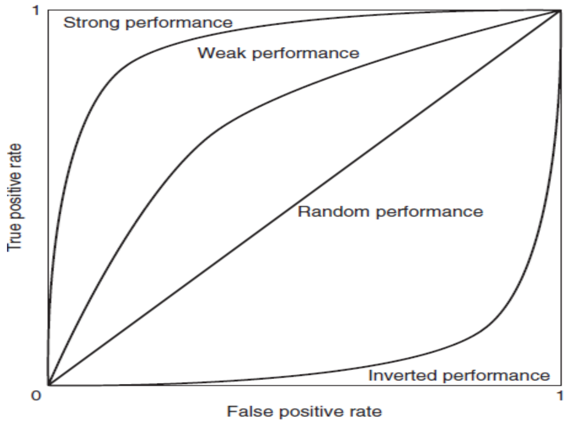


Figure 2.4 Illustration of ROC Curve (Source: (H. He & Ma, 2013))

Area under ROC curve (AUC) is also a commonly preferred metric which is derived from ROC curve. It is helpful especially when dealing with an imbalanced dataset.

AUC is the ability of a classifier to distinguish between classes. For binary classification AUC is formulated as follows (Hossin & Sulaiman, 2015):

$$AUC = \frac{S_p - n_p(n_n + 1)/2}{n_p \times n_n} \tag{2.47}$$

where S_p is the sum of all positive observations ranked, n_p and n_n denote the number of positive and negative observations. Higher AUC means better differentiation between classes. One drawback of AUC is computation time for multiclass classification (Hossin & Sulaiman, 2015).

Anaklı (2009) compares the prediction and classification methods and selects the one for the quality problem in her M. Sc. thesis study. For this aim, she first applies Analytical Network Process (ANP) to find weights for sub-criteria then applied the Preference Ranking Organization Method for Enrichment Evaluation (PROMETHEE) to rank methods. Anaklı (2009) uses a comprehensive list of criteria to find the best method among all machine learning methods. The list involves quantitative and qualitative criteria. For quantitative criteria, Anaklı (2009) uses common prediction and classification evaluation metrics such as, accuracy, f1-score, precision, recall etc. In order to reduce this list and select the best metrics among them, Anaklı (2009) conducts a correlation and factor analysis of these metrics. At the end, highly correlated measures are eliminated and for each group one measure is selected to represent. In the final sub-criteria list, misclassification rate (MCR), precision, recall, kappa, stability and AUC are selected for the PROMOTHEE application.

In the literature, for different types of studies the performance of XGBoost and Random Forest is compared by researchers. However, selection of the performance metric is not only dependent on the algorithms, in fact it actually depends on the nature of the data, reason of applying the classification problem and what is trying to be obtained at the end of the algorithm. In Table 2.11. classification metrics and the papers which compare Random Forest and XGBoost prefer to use them are summarized.

Table 2.11 Classification metrics used to compare Random Forest and XGBoost

Metric	**References**
Accuracy	Kabiraj et al. (2020), Bentéjac et al., (2020), Jhaveri et al., (2019), Quevedo. et al., (2021), Rabby et al., (2020), Bentéjac et al., (2020), Senthan et al., (2021), Jhaveri et al., (2019), Rabby et al., (2020), Quevedo. et al., (2021)
Recall	Kabiraj et al. (2020), Xie et al., (2021), Meng et al. (2018), Senthan et al., (2021),Jhaveri et al., (2019), Rabby et al., (2020), Quevedo. et al., (2021), Anaklı, (2009)
Precision	Kabiraj et al. (2020), Xie et al., (2021), Meng et al. (2018), Senthan et al., (2021), Jhaveri et al., (2019), Anaklı, (2009)
F-1 Score	Kabiraj et al. (2020), Xie et al., (2021), Senthan et al., (2021)
AUC	Xie et al., (2021), Kidando et al., (2021), Bentéjac et al., (2020), Çelik and Osmanoğlu, (2019), Quevedo. et al., (2021), Anaklı, (2009)
Kappa	Rabby et al., (2020), Anaklı, (2009)
Balanced Accuracy	Kidando et al., (2021)
Specificity	Kabiraj et al. (2020), Rabby et al., (2020), Quevedo. et al., (2021)

Many researchers preferred accuracy to compare XGBoost and Random Forest, followed by precision, recall and AUC. However, in the case studies data is imbalanced. Using accuracy can be misleading. To obtain robust results, appropriate metrics for imbalance should be selected. Therefore, it is preferred to use precision, recall and AUC to compare performance of data mining algorithms with MCDM methods.

2.3.10 Cross-Validation

Cross-validation is a resampling method to test if models' performance is valid for an independent dataset. In k-fold cross validation, the dataset divided into k equal distinct subset. In each step, one of the $k-1$ subsets are used for training and the remaining subset is used for testing purposes. Totally, performance of estimator measured k times (Hand, 2007). The structure of k-fold cross-validation is shown in Figure 2.5.

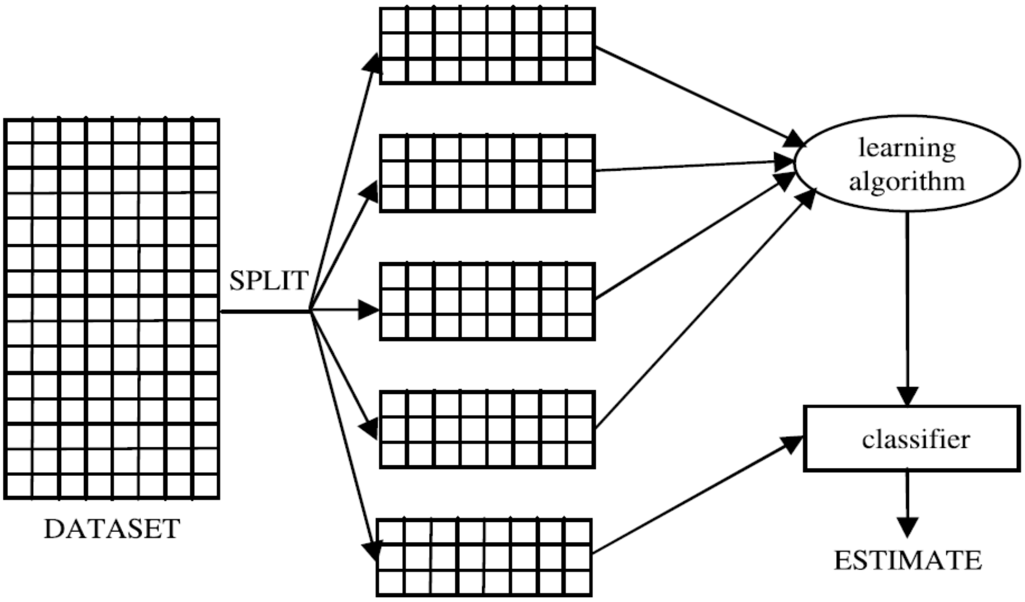


Figure 2.5 k-fold Cross Validation (Source: Hand, (2007))

However, it is better to use stratified k-fold cross validation when the dataset is imbalanced (He & Ma, 2013). Stratified k-fold cross validation preserves the same distribution of the original dataset in each fold; therefore, the possibility of not showing a rare class in any fold is eliminated.

2.3.11 Grid Search Cross-Validation

Grid Search cross-validation (GSCV) is a method that finds the best parameter setting for an estimator using all combinations of specified parameter space with the cross-validation method with a given performance metric (Buitinck, et al., 2013).

2.4 Multi-Criteria Decision-Making Methods

Multi-criteria decision making (MCDM) is a collection of methods which help Decision Maker (DM) to make decisions under a conflicting set of criteria and objectives. MCDM has a wide range of application areas such as supplier selection, customer relationship management, flood risk analysis, financial analysis, and disaster management (Ishizaka & Nemery, 2013).

Discrete-alternative MCDM methods can be chosen in cases where a set of alternatives is known and integer valued (Haddad & Sanders, 2018). The most common approaches are as follows.

- Analytic Hierarchy Process (AHP)
- Analytic Network Process (ANP)
- Technique of Order Preference Similarity to the Ideal Solution (TOPSIS)
- VIKOR
- Preference ranking organization method for enrichment of evaluations (PROMETHEE)
- ÉLimination et Choix Traduisant la REalité (ELECTRE)

- Multi-attribute utility theory (MAUT)
- Multi-Objective Optimization by Ratio Analysis (MOORA)

AHP is one of the simplest discrete MCDM methods. It divides problem into hierarchy levels and constructs a decision tree in this base (Balali et al., 2014). There are basically three levels of the hierarchy; goal of the problem, criteria to define alternatives and alternatives at the lowest level. Pairwise comparisons are conducted at each level with respect to the higher level.

PROMETHEE (Preference Ranking Organization Method for Enrichment of Evaluations) is in the family of outranking methods proposed by Brans and Vincke (1984). There are various versions of PROMETHEE proposed in the literature, but two common versions are PROMETHEE I and PROMETHEE II. PROMETHEE II provides a full ranking of alternatives using aggregated net preference flow (Balali et al., 2014; Opricovic & Tzeng, 2007). In PROMETHEE I, pairwise comparisons are conducted to compare each alternative. There are some additional parameters to be defined at first by the DMs. Each criterion weight is determined appropriately. The preference functions are used to define the preference degree between each pair of alternatives for each criterion between range [0,1]. There are six types of preference function. (1) usual, (2) U-shape, (3) V-shape, (4) level, (5) V-shape with indifference and (6) Gaussian. Also, preference threshold, p, and indifference threshold, q, need to be defined for each criterion. (Balali et al., 2014; Opricovic & Tzeng, 2007). Balali et al. (2014) indicates that selecting an appropriate preference function can be very difficult for DMs who do not have enough experience. In PROMETHEE, first using the preference function, pairwise comparisons are conducted to find deviations of each criterion for each pair of alternatives. In the next step, the global preference index is calculated for each pair and entering and leaving flows are calculated and partial ranking is obtained. Entering flow is the metric which shows how an alternative dominates the others. Similarly, leaving flow shows how an alternative is dominated by the other alternatives. At the end, aggregating leaving and entering flows, net flow is obtained. (Balali et al., 2014; Pohekar &

Ramachandran, 2004). Anaklı (2009) uses the PROMETHEE in her thesis study. The weights of the sub-criteria are obtained with the Analytic Hierarchy Process (AHP). For quantitative criteria, thresholds are determined with DMs, literature, and Repeated-ANOVA. However, selection of $p, v,$ and the preference function needs a DM who has sufficient experience and knowledge, therefore this method is not selected for this study.

The ELECTRE method is proposed by Roy (1996). It involves various alternative versions (I, II, III, IV). One of the most widely used one is ELECTRE II for outranking. It is based on the idea of concordance and discordance (Opricovic & Tzeng, 2007). Like in PROMETHEE, it also uses pairwise comparisons. Strong and weak relationships are identified, and global concordance is used to prove that an alternative outranks its pair over all criteria. The concordance, discordance indices and thresholds are the parameters of the method which are decided by DM. The relationship between each pair of alternatives is explained with the concordance and discordance values. Pohekar and Ramachandran (2004) mentions that sometimes ELECTRE is not able to find preferred alternatives. It is appropriate for the case when there are fewer criteria and a relatively large number of alternatives.

Multi-attribute utility theory (MAUT) uses utility functions to express DM’s preference. Utility function is the combination of a set of attributes that can be defined additively or multiplicatively (Pohekar & Ramachandran, 2004).

TOPSIS and VIKOR are the MCDM methods based on distances. In TOPSIS, the best solution is the one that is closest to the ideal solution and furthest to the anti-ideal solution. All information about criteria is gathered in the decision matrix $|\boldsymbol{X}|$= (x_{ij}) where $i = 1, \dots, m$ and $j = 1, \dots, n.$ m and n denote the number of alternatives and the number of criteria, respectively. The criteria can be in different units (years, dollar, kg etc.) (Ceballos et al., 2016; Ishizaka & Nemery, 2013). The method consists of five steps as explained below.

Step 1: Decision matrix is normalized in order to compare different units.

$$n_{ij} = \frac{x_{ij}}{\sqrt{\sum_{a=1}^{m} x_{aj}^2}} \text{ where } i = 1, \dots, m \; and \; j = 1, \dots, n. \qquad (2.48)$$

Step 2: Weighted normalized decision matrix obtained.

$$\boldsymbol{v_{ij}} = w_i \times n_{ij} \qquad (2.49)$$

Step 3: Calculate ideal and anti-ideal solutions

$$\text{Ideal Solution} = \boldsymbol{A}^{+} = \{ v_1^{+}, v_2^{+}, \dots, v_j^{+}, \dots, v_n^{+} \} \qquad (2.50)$$

$$\text{Anti-ideal Solution} = \boldsymbol{A}^{-} = \{ v_1^{-}, v_2^{-}, \dots, v_j^{-}, \dots, v_n^{-} \} \qquad (2.51)$$

where $v_j^{+} = max_i(v_{ij})$ and $v_j^{-} = \min_i(v_{ij})$ if the j^{th} criterion is benefit, otherwise $v_j^{+} = \min_i(v_{ij})$ and $v_j^{-} = max_i(v_{ij}),\; i = 1,2, \dots, m$ and $j = 1, 2, \dots, n.$

Step 4: Calculate the distances from ideal and anti-ideal solutions for each alternative.

$$d_i^{+} = \sqrt{\sum_{j=1}^{n} (v_i^{+} - v_{ij})^2}\,, i = 1, \dots, m \qquad (2.52)$$

$$d_i^{-} = \sqrt{\sum_{j=1}^{n} (v_i^{-} - v_{ij})^2}\,, i = 1, \dots, m \qquad (2.53)$$

Step 5: Calculate the relative closeness coefficient for each alternative.

$$C_i = \frac{d_i^{-}}{(d_i^{-} + d_i^{+})} \qquad (2.54)$$

where C_i is in range $[0, 1]$. Preferred solution is the alternative with the highest relative closeness coefficient (Ishizaka & Nemery, 2013).

There are some differences between TOPSIS and VIKOR pointed out by Opricovic and Tzeng (2004). The TOPSIS method use L_2 metric to calculate C_i, while the VIKOR method uses L_1 and L_∞ to calculate the final ranking. Moreover, the TOPSIS method uses vector normalization, and the VIKOR method uses linear normalization. One advantage of VIKOR is being appropriate for DM who has not experiences. Both methods provide ranks as result. In VIKOR, alternative which has the highest rank, is the one closest to the ideal solution, but this may not be case in TOPSIS (Opricovic & Tzeng, 2004).

MOORA is a combination of multiple MODM methods: Ratio System and Reference Point. Multi-MOORA also uses Full Multiplicative Form. The general steps of the method are given in Figure 2.6.

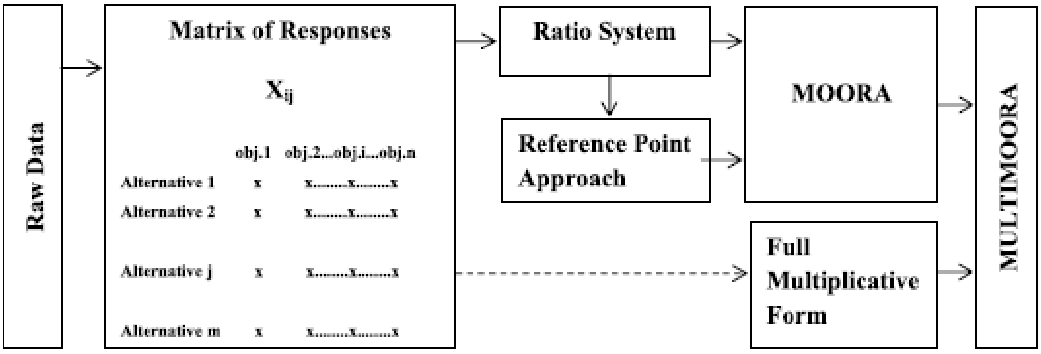

Raw Data
Matrix of Responses
X_{ij}
obj.1 obj.2...obj.i...obj.n
Alternative 1 x x.........x.........x
Alternative 2 x x.........x.........x
Alternative j x x.........x.........x
Alternative m x x.........x.........x
Ratio System
Reference Point Approach
MOORA
Full Multiplicative Form
MULTIMOORA

Figure 2.6 Diagram of Multi-MOORA (Source: (Brauers & Zavadskas, 2012))

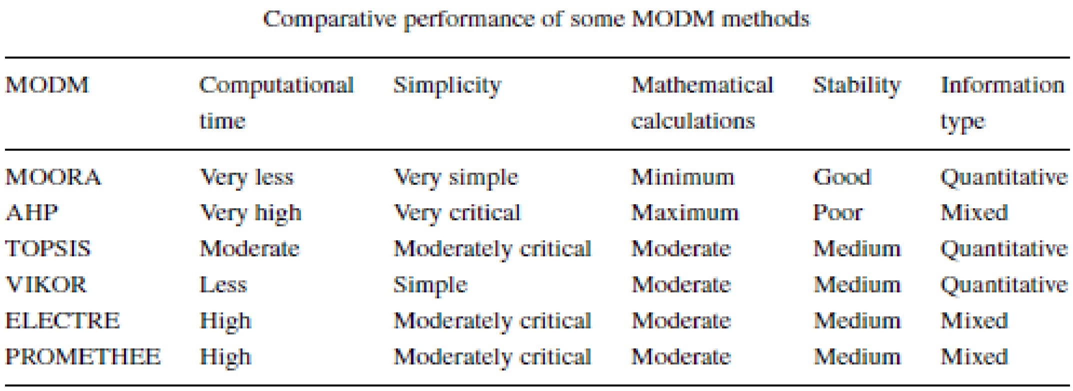
Brauers & Zavadskas (2012) mentions conditions that a robust multi-objective method should satisfy as follows.

1. A multi-objective method is more robust if all decision makers interested in that problem involve the process.
2. A multi-objective method which considers all non-correlated objectives is more robust than others.
3. A multi-objective is more robust than others if all interrelations between objectives and alternatives are taken into account as a whole rather than the one with compare alternatives as pairs.
4. The method uses non-subjective methods for choice of objectives, parameter selection, normalization and giving importance to objectives is more robust than the others.

5. The method based on cardinal numbers is more robust than the one using ordinal numbers.

6. The multi-objective method which uses recent data is more robust than the one that uses old data.

7. The method which satisfies these six conditions and uses more than one multi-objective method is more robust than the one uses only one multi-objective method. Robustness increases with the increased number of multi-objective methods added to the process. They said that MOORA satisfies these conditions. TOPSIS and VIKOR violate the first condition and Euclidean distance in TOPSIS and rectangular distance in VIKOR deviates from real ranking.



Comparative performance of some MODM methods

MODM	Computational time	Simplicity	Mathematical calculations	Stability	Information type
MOORA	Very less	Very simple	Minimum	Good	Quantitative
AHP	Very high	Very critical	Maximum	Poor	Mixed
TOPSIS	Moderate	Moderately critical	Moderate	Medium	Quantitative
VIKOR	Less	Simple	Moderate	Medium	Quantitative
ELECTRE	High	Moderately critical	Moderate	Medium	Mixed
PROMETHEE	High	Moderately critical	Moderate	Medium	Mixed

Figure 2.7 Comparison of Multi-Objective Decision Making Method (Source: (Brauers & Zavadskas, 2012))

They compare MODM methods based on computational time, simplicity, mathematical calculations, stability, and information type as shown in Figure 2.7. Based on this comparison MOORA seems a better approach than the others.

Multi-MOORA applies vector normalization to generate comparable ratings. Hafezalkotob et al. (2019) mentions that TOPSIS and VIKOR are goal or reference level models. Both use l_p -metrics. TOPSIS has two reference points: positive ideal solution and negative ideal solution. However, Reference Point Approach sometimes cannot differentiate alternatives and give them the same rankings. For Multi-

MOORA the Reference Point Approach uses $Min - Max\ Metric$ of Tchebycheff, which is calculated as follows.

$$r_j = \left\{ \max_i x_{ij}^*, \quad j \leq g;\ \min_i x_{ij}^*, \quad j > g \ \right\} \tag{2.55}$$

$$d_{ij} = \left| w_j r_j - w_j x_{ij}^* \right| \tag{2.56}$$

$$z_i = \max_j d_{ij} \tag{2.57}$$

where g is the number of beneficial criteria. The best alternative with this method has the minimum utility z_i, the ranking is obtained in ascending order (Hafezalkotob et al., 2019).

Ratio System uses arithmetic weighted aggregation operator and an alternative with poor performance on some criteria and good performance in some other criteria may be preferred to an alternative that has moderate performance on all criteria. The calculations of Ratio System are given below.

$$x_{ij}^* = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{m} x_{ij}^2}} \tag{2.58}$$

where x_{ij} is the response of alternative i on objective j; $i = 1, \ldots, m$; m is the number of alternatives; $j = 1, \ldots, n; n$ is the number of objectives.

$$y_i^* = \sum_{j=1}^{g} s_i x_{ij}^* - \sum_{j=g+1}^{n} s_i x_{ij}^* \tag{2.59}$$

With $i = 1, \ldots, g$ as the objectives to be maximized; $i = g + 1, \ldots, n$ as the objectives to be minimized. s_i is the significance coefficient of objective i and y_{ij}^* is the total assessment with significance coefficients of alternative j with respect to all objectives. The alternative with the highest value ranked as 1st.

Full multiplicative form is an incompletely compensatory model and uses the geometric weighted aggregation. The utility obtained with the product of weighted normalized alternatives on beneficial criteria is divided by the product of weighted alternatives on cost criteria. Results are sorted in decreasing order, the alternative with the highest utility ranked as best.

$$u_i = \frac{\prod_{j=1}^{g}\left(x_{ij}^{*}\right)^{w_j}}{\prod_{j=g+1}^{n}\left(x_{ij}^{*}\right)^{w_j}} \tag{2.60}$$

With Reference Point Approach, Ratio System and Full Multiplicative Form, subordinate rankings are obtained and combined to obtain a more robust ranking list than individual ranking. In that part, dominance theory is used in the original Multi-MOORA and preferred by researchers mostly.

Ceballos et al. (2016), compare VIKOR, TOPSIS and Multi-MOORA. Their experimental setup consists of randomly generated MCDM problems, totally 1600. They found that VIKOR is sensitive to parameter setting, and Multi-MOORA and TOPSIS produce very similar results. Also, advantages of Multi-MOORA are listed as, less computational time, easy to implement and produce stabilized quantitative results (Chakraborty, 2011; Hafezalkotob et al., 2019). Multi-MOORA and TOPSIS are used to compare performances of different data mining algorithms in this study.

CHAPTER 3

PROPOSED METHOD

For RPD studies with ordinal response, many data mining algorithms are used to analyze data. One of them is Logistic Regression (Erdural, 2006; Erişkin et al., 2021; Karabulut, 2013). Gülbudak (2018) proposes a method for RPD with ordinal response and uses Random Forest. In this study, XGBoost is also taken into consideration because of its good features as mentioned in the literature review. After tuning the hyperparameters of the methods for four data sets, the performance of these algorithms is compared with TOPSIS and Multi-MOORA. A sensitivity analysis is also performed for TOPSIS. As a result, XGBoost and Random Forest have better performance than Logistic Regression. Since Random Forest is easier to implement, it is chosen for the rest of the analysis.

Then, the proposed method for RPD with a categorical response, flowchart of which is given in Figure 3.25, is explained. The method predicts category probabilities at each experimental trial using RF and calculates median and COV at each experimental trial using these probabilities. Empirical models of the median and COV are built as functions of design parameters. Using these empirical models, a nonlinear optimization model is solved to find the levels of design parameters that produce desired median value and minimize the COV. The method is applied to three different case studies. One of them is a smaller-the-better type of problem and others are larger-the-better type of problems, all have multiple classes.

3.1 Comparison of Data Mining Algorithms

Four different datasets are used to compare the performance of Random Forest, Logistic Regression and XGBoost. Before starting the comparison, the hyperparameters of each algorithm are tuned. Afterwards, repeated stratified 3-fold cross validation is applied to algorithms. Then, TOPSIS and Multi-MOORA are applied to the average of test results for precision, recall and AUC.

XGBoost (XGB), Random Forest (RF), and Logistic Regression (LR) are applied to four experimental datasets. In this section, how to tune hyperparameters of each data mining algorithm, and comparison of performance of each model are explained. Moreover, the best model is selected for each of these datasets applying cross validation and MCDM methods.

3.1.1 Surface Defect Data

The detailed explanation of the dataset for surface defect case study is provided below.

When there is a skewed distribution between classes in the dataset, an imbalanced data problem occurs. Brownlee (2020) categorizes imbalance data as slightly imbalance where number of observations between majority and minority classes are different in a small amount (e.g. 5:8), and severe imbalance where there is a high degree of difference between majority and minority classes (1:100 or more). Imbalance ratio is defined as the number of instances in the largest class divided into the number of instances in the smallest class (Mullick et al., 2020).

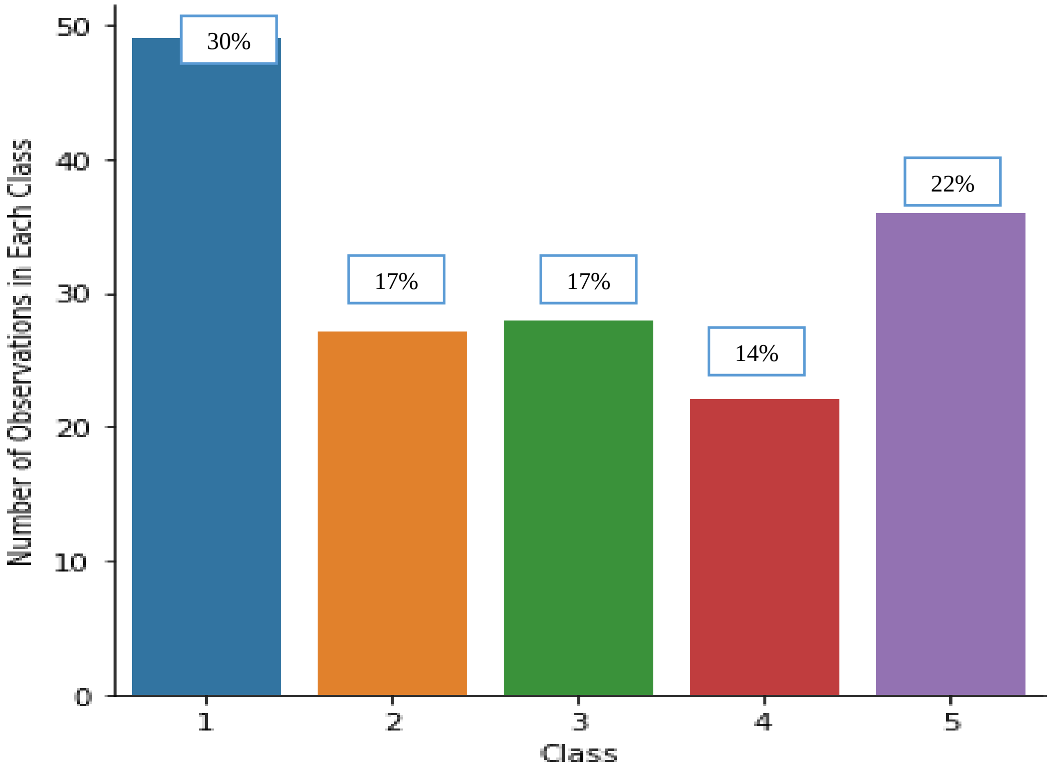


Figure 3.1 The Distribution of Classes in Surface Defect Data

The distribution of the classes is shown in Figure 3.1, 30% of the observations belong to Class I. The distribution of observations in other classes are 17%, 17%, 14%, 22%, respectively. It can be said that the data is slightly imbalanced. Therefore, it is preferred to use a repeated stratified 3-fold cross validation in the GSCV method to reduce the imbalance effect.

To obtain the best parameters, the GSCV and model specific methods are used. Performance criteria are selected as AUC for RF because all the datasets are imbalanced, and log loss for XGB since XGBoost uses log loss function to optimize trees. To select parameter space for Random Forest and XGBoost, a similar setting is applied as Ceballos et al. (2016) uses. These values are given in Table 3.1.

Table 3.1 Surface Defect Data, Grid Search Values for XGB, RF, LR for Surface Defect Data

Model	Grid Search Values	Best Parameters	Performance Metric	Best Value
XGB	$subsample$: $[0.5, 0.75, 1]$, $colsample\_bytree$: $[0.5, 0.75, 1]$, $learning\_rate$: $[0.01, 0.05, 0.1]$, max_depth: $[2, 4, 6]$, $gamma$: $[0.1, 0.3, 0.5, 1, 2]$, max_delta_step: $[1, 2]$, $scale\_pos\_weight$: $[1, 3, 5]$	$subsample$: 0.5, $colsample\_bytree$: 0.75 $learning\_rate$: 0.05, max_depth: 4, $gamma$: 0.5, max_delta_step: 2, $scale\_pos\_weight$: 1	Log Loss	1.19
RF	$max\ \_features$: $[2, 3, 4, 5,6]$, $n\_estimators$: $range(100, 10{,}000)$	$max\ \_features$: 4, $n\_estimators$: 1400	AUC	79%
LR	Default is used	-	-	-

3.1.1.1 XGBoost Tuning

Using the best parameters in Table 3.1, the number of estimators is found with the built-in cross validation in XGBoost (XGBoost Developers, 2020). The cross-validation results are represented for both Log Loss and AUC metrics in Figures 3.2 and 3.4. Different numbers of trees are tried between 100 to 1,000.

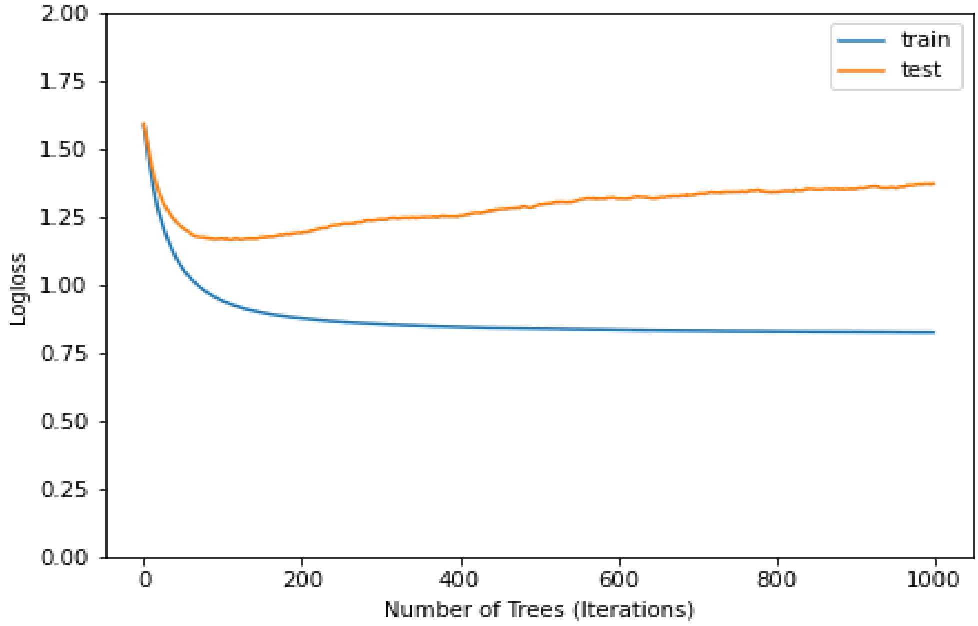


Figure 3.2 XGB, The Number of Trees versus Log Loss for Surface Defect Data

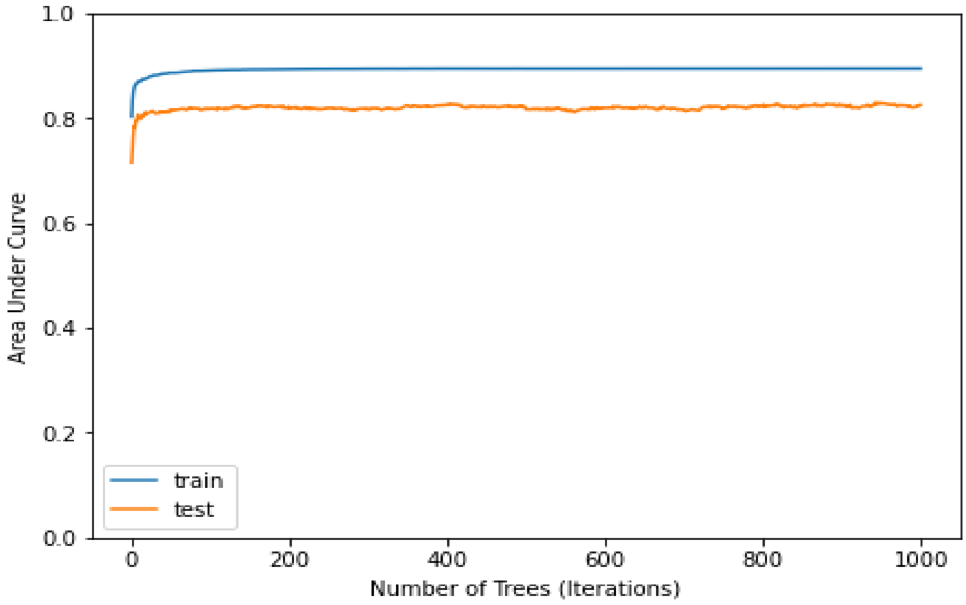


Figure 3.3 XGB, The Number of Trees versus AUC for Surface Defect Data

In Figure 3.3, the gap between train and test data increases after 100 iterations (trees). Selecting the number of trees as a high number may cause an overfitting problem. Also, AUC results do not change after 100 iterations. Therefore, the number of trees for XGB model is selected 100.

3.1.1.2 Random Forest Tuning

Both GSCV as mentioned above and OOB error are used to tune the Random Forest model to double check results. GSCV results are provided in Table 3.1. The two important hyperparameters of Random Forest are tuned using OBB error. The obtained results are given in Figures 3.4 and 3.5.

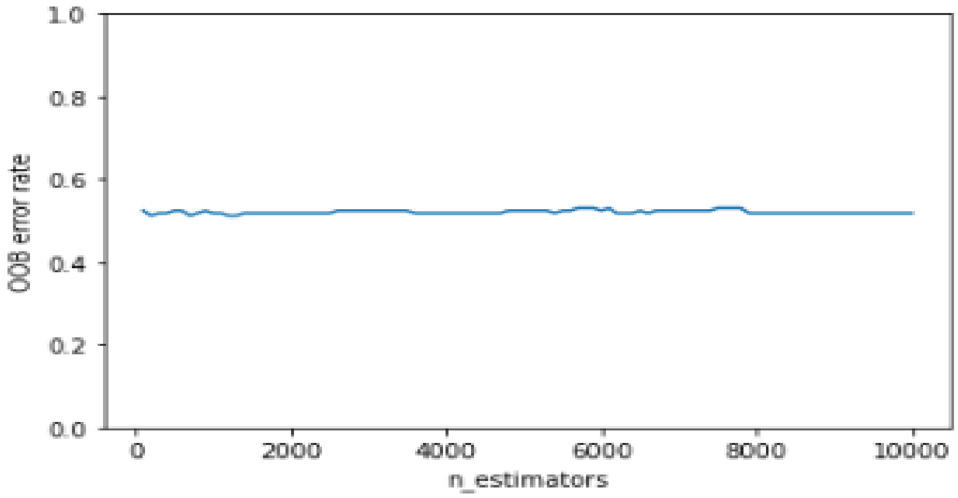


Figure 3.4 RF $n\_estimators$ versus OOB error for Surface Defect Data

The lowest OOB error is 0.518 for RF where number of estimators is 1400. This result is also validating what is obtained with the GSCV results. So, the number of estimators (trees) for RF is selected as 1400. OOB error rate is constant, so GSCV results are used as $max\ \_features$ in this model.

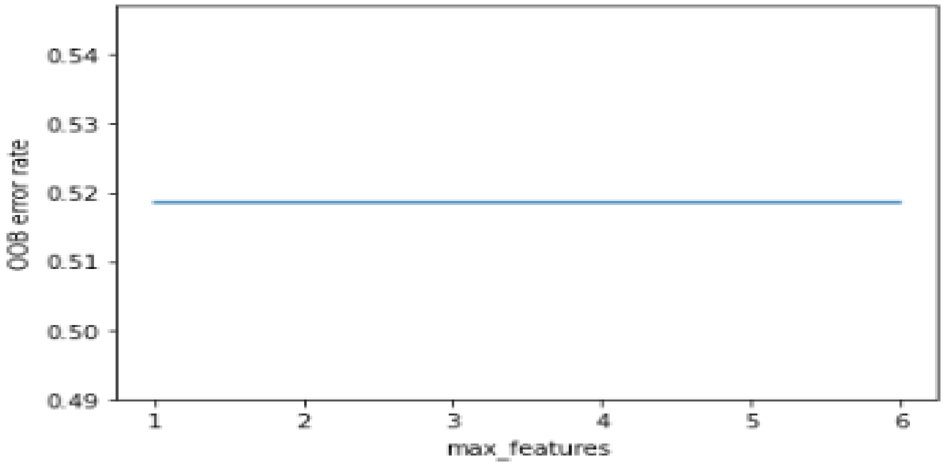


Figure 3.5 RF $max\ \_features$ versus OOB error for Surface Defect Data

OOB error rate is constant, so GSCV results are used as $max\ \_features$ in this model.

3.1.1.3 Performance Results of Models

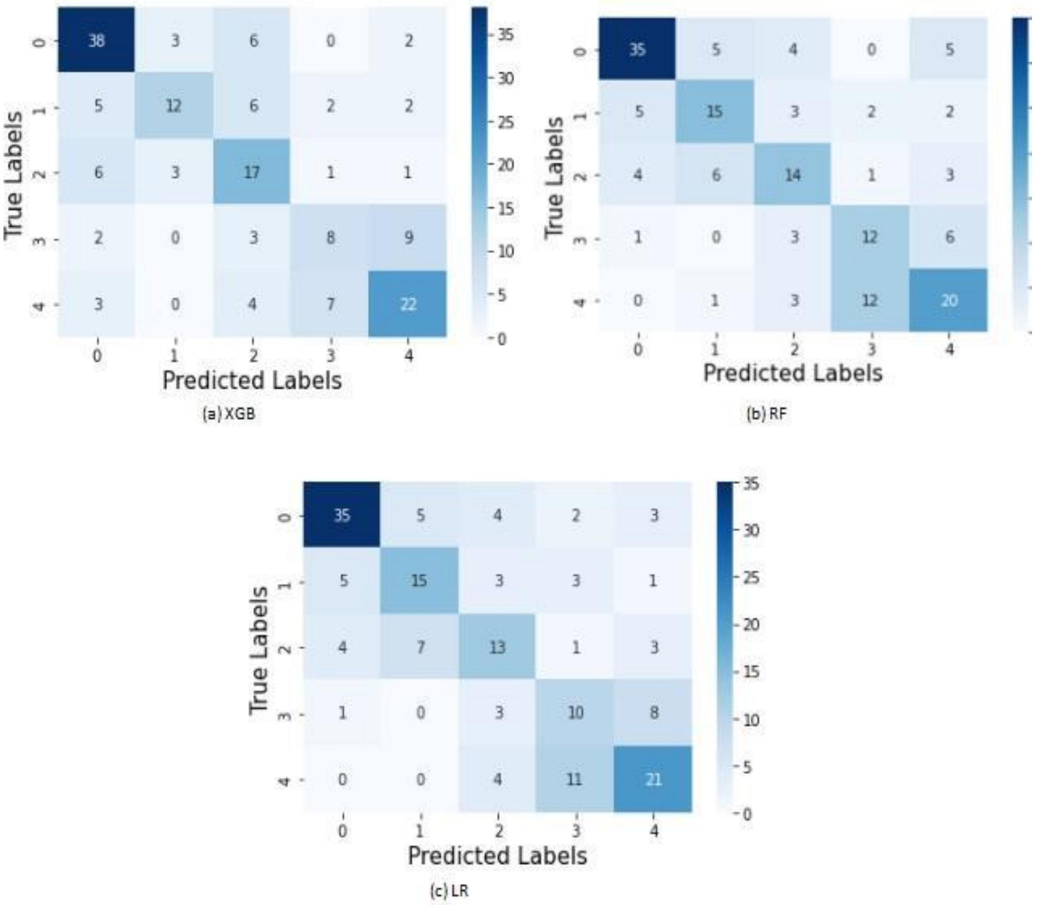
After using these hyperparameters, models are fitted to surface defect data and obtained performance results are given in Table 3.2, Table 3.3, and in Figure 3.6.

Table 3.2 Performance Results for XGB, RF, RF_B, LR for Surface Defect Data

Model	**Accuracy**	**F1-score**	**Balanced Accuracy**	**Kappa**	**G-mean**	**AUC**
XGB	**60%**	**60%**	56%	**48%**	54%	**88%**
RF	59%	59%	**57%**	**48%**	**57%**	**88%**
LR	58%	58%	55%	47%	55%	83%

Table 3.3 Performance Results for XGB, RF, RF_B, LR for Surface Defect Data

Model	Recall					Precision					Specificity				
	Class I	**Class II**	**Class III**	**Class IV**	**Class V**	**Class I**	**Class II**	**Class III**	**Class IV**	**Class V**	**Class I**	**Class II**	**Class III**	**Class IV**	**Class V**
XGB	78%	44%	61%	36%	61%	70%	67%	47%	44%	61%	89%	96%	86%	93%	89%
RF	71%	56%	50%	55%	56%	78%	56%	52%	44%	56%	91%	91%	90%	89%	87%
LR	71%	56%	46%	45%	58%	78%	56%	48%	37%	58%	91%	91%	90%	88%	88%



(a) XGB

(b) RF

(c) LR

Figure 3.6 Confusion Matrices (a) XGB, (b) RF, (c) LR for Surface Defect Data

In overall, no method is clearly having better results. For accuracy and F-1 score, XGB has the best performance while for balanced accuracy and g-mean, RF shows better performance than the others. AUC is the same for XGB and RF methods. LR results are slightly worse than XGB and RF. Selecting the best method considering these results can be misleading in this stage.

3.1.2 Inkjet Printer Data

The detailed explanation about the dataset is given in the subsequent paragraphs.

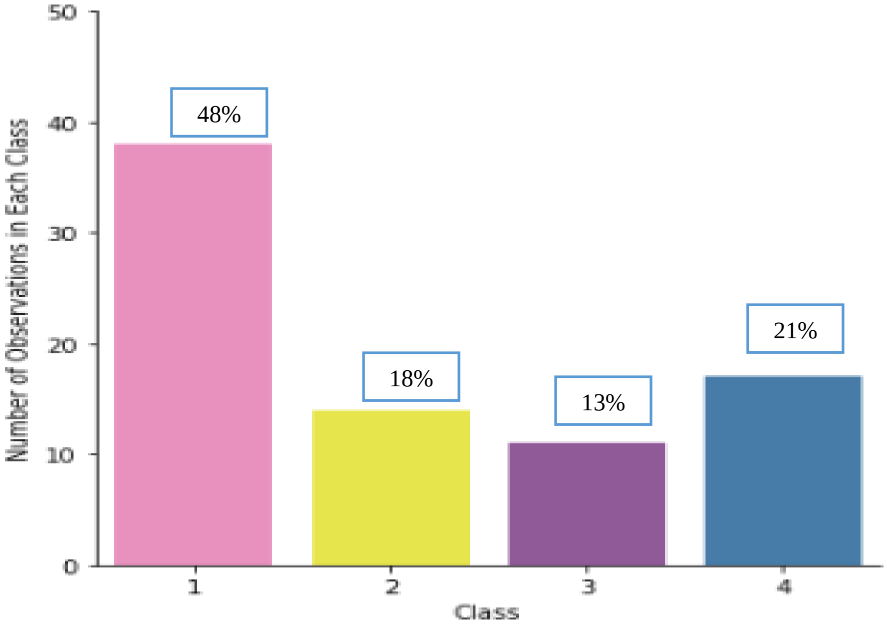


Figure 3.7 Class Distribution of Inkjet Printer Data

As it can be seen in Figure 3.7, Class I has most of the data. 48% of the data belongs to Class I and 18%, 13%, 21% of the data belongs to other classes, respectively. Based on these, it can be said that inkjet printer data is slightly imbalanced. Similar to Surface Defect data, imbalanced data problem is handled using stratified 3-fold cross validation in Grid Search method. The selected hyperparameters are provided in Table 3.4.

Table 3.4 Inkjet Printer Data, Grid Search Values for XGB, RF, LR

Model	Grid Search Values	Best Parameters	Performance Metric	Best Value
XGB	$subsample: [0.5, 0.75, 1],$ $colsample\_bytree: [0.5, 0.75, 1],$ $learning\_rate: [0.01, 0.05, 0.1],$ $max\_depth: [2, 4, 6],$ $gamma: [0.1, 0.3, 0.5, 1, 2],$ $max\_delta\_step: [1, 2],$ $scale\_pos\_weight: [1, 3, 5]$	$subsample: 0.5,$ $colsample\_bytree: 0.75$ $learning\_rate: 0.05,$ $max\_depth: 4,$ $gamma: 0.3,$ $max\_delta\_step: 2,$ $scale\_pos\_weight: 1$	Log Loss	1.12
RF	$max\ \_features: [2, 3, 4, 5],$ $n\_estimators: range(100, 10{,}000)$	$max\ \_features: 2,$ $n\_estimators: 300)$	AUC	70%
LR	Default is used	-	-	-

3.1.2.1 XGBoost Tuning

The best parameters in Table 3.4 are used to find the number of estimators ($n\_estimators$) for XGB. The cross-validation results are represented for both Log Loss and AUC metrics in Figures 3.8 and 3.9. Different numbers of trees are tried between 100 to 1,000.

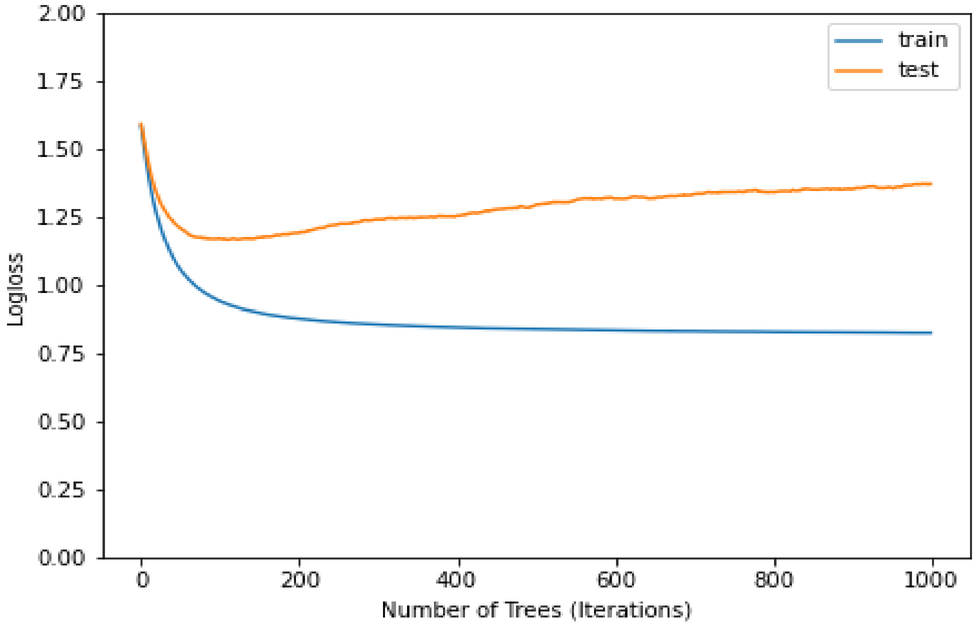


Figure 3.8 XGB, The Number of Trees versus Log Loss for Inkjet Printer Data

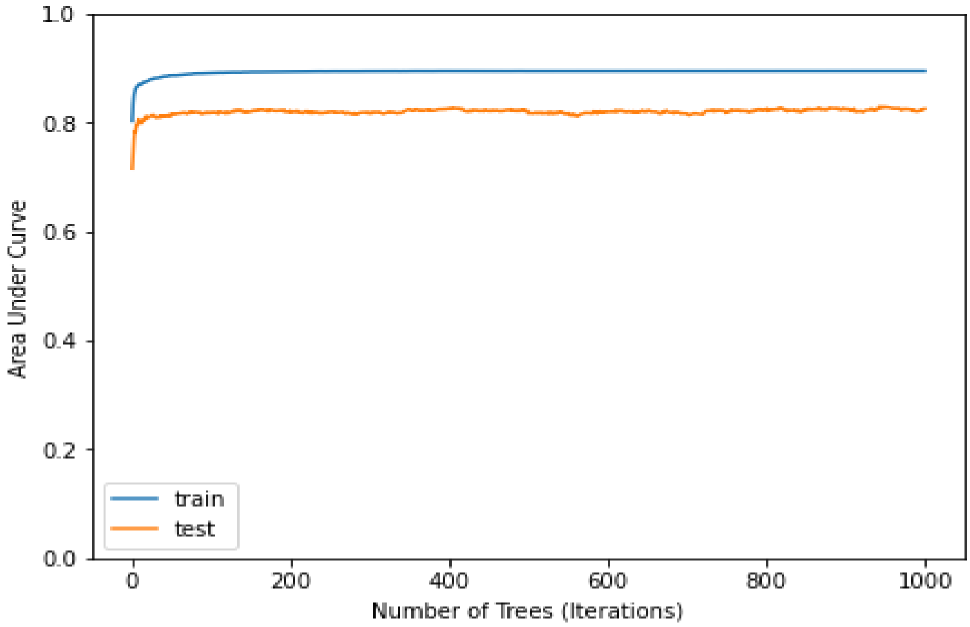


Figure 3.9 XGB, The Number of Trees versus AUC for Inkjet Printer Data

AUC is at satisfactory level and stable after 100 iterations. Also, as the number of iterations increase, the gap between training and test results grows for Log Loss. Considering all these, 100 is selected as the number of trees.

3.1.2.2 Random Forest Tuning

GSCV results are provided in Table 3.4. The two important hyperparameters of Random Forest are tuned using OBB error. The obtained results are presented in Figures 3.10 and 3.11.

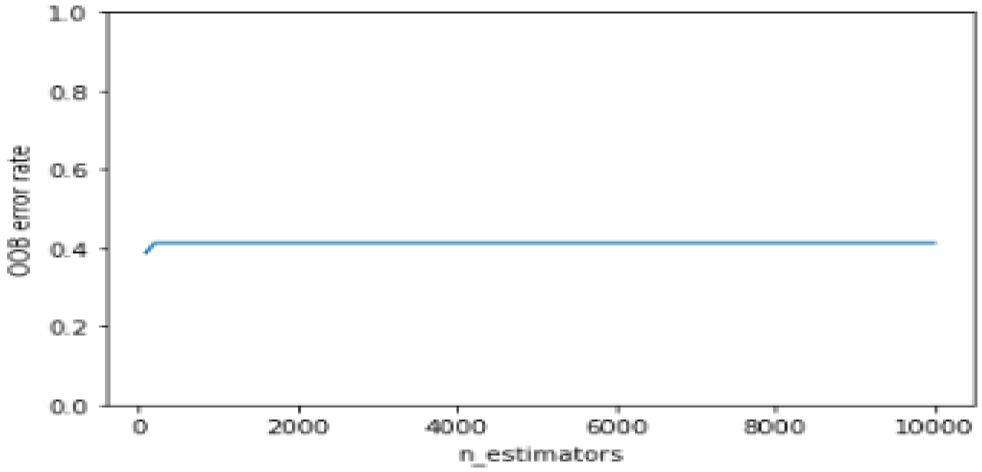


Figure 3.10 RF $n\_estimators$ versus OOB error for Inkjet Printer Data

OOB error is 0.387 where the number of trees is 100. However, it is 0.43 for the GSCV result where number of trees is 300. This result conflicts with GSCV due to the fact that the GSCV and OOB estimates consider different performance criteria. While the GSCV tries to maximize AUC, in OOB estimates, error rate is tried to be minimized. Since OOB error provides the unbiased estimate of the true prediction error, $n_estimator$s is chosen as 100 for RF.

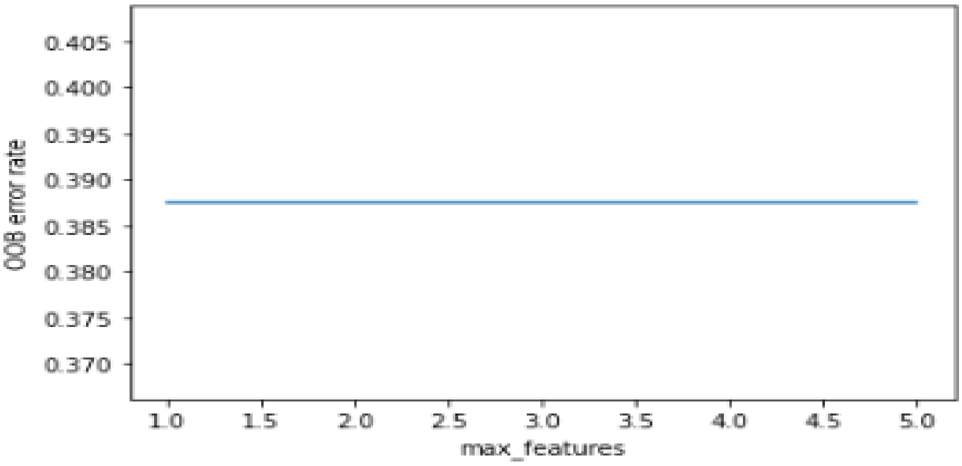


Figure 3.11 RF $max\ \_features$ versus OOB error for Inkjet Printer Data

OOB error rate is constant, so GSCV results are used as $max\ \_features$ for the model.

3.1.2.3 Performance Results of Models

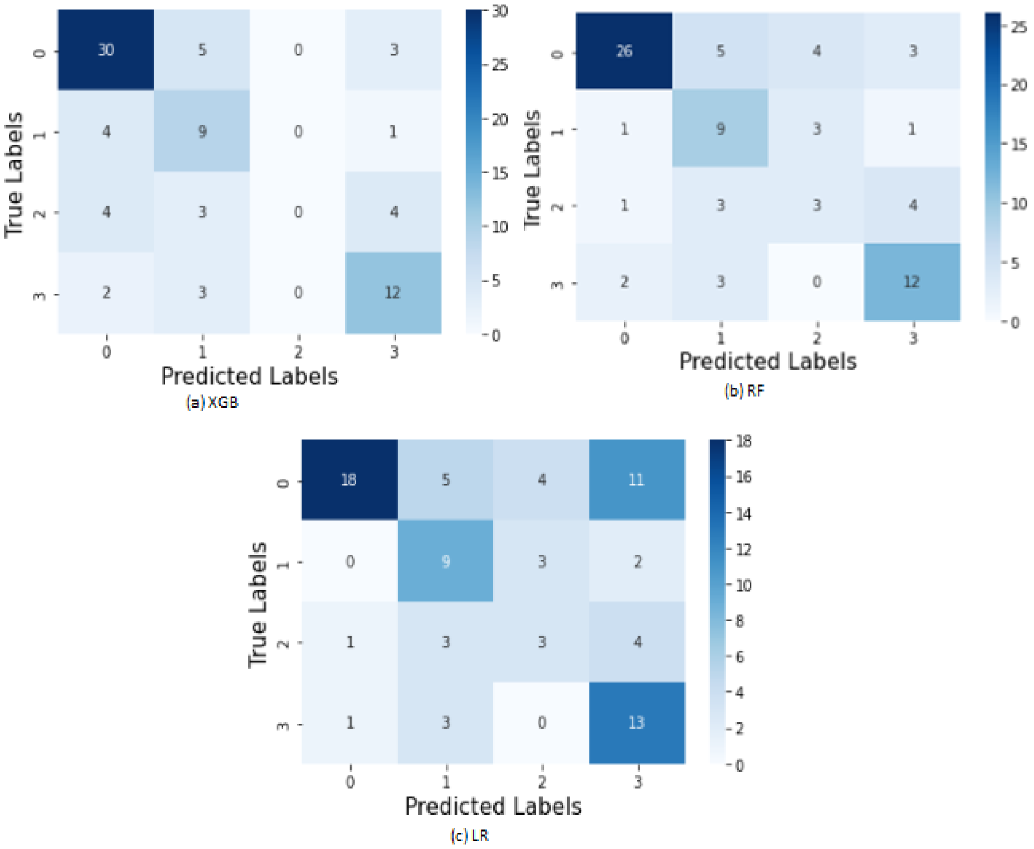
In Table 3.5, Table 3.6, and Figure 3.12, overall performance results for XGB, RF, and LR are provided. For accuracy and f1-score, XGB has the best performances while for balanced accuracy kappa, and g-mean, RF shows better performance than the others. LR results are worse than XGB and RF.

Table 3.5 Performance Results for XGB, RF, LR for Inkjet Printer Data

Model	**Accuracy**	**F1-score**	**Balanced Accuracy**	**Kappa**	**G-mean**	**AUC**
XGB	**64%**	**64%**	53%	46%	0	**83%**
RF	62%	62%	**58%**	**47%**	**54%**	**83%**
LR	54%	54%	54%	38%	50%	77%

Table 3.6 Recall, Precision, and Specificity Results for XGB, RF, LR for Inkjet Printer Data

Model	Recall				Precision				Specificity			
	Class I	**Class II**	**Class III**	**Class IV**	**Class I**	**Class II**	**Class III**	**Class IV**	**Class I**	**Class II**	**Class III**	**Class IV**
XGB	79%	64%	0%	71%	75%	45%	0%	60%	76%	83%	100%	87%
RF	68%	64%	27%	71%	87%	45%	30%	60%	90%	83%	90%	87%
LR	47%	64%	27%	76%	90%	45%	30%	43%	95%	83%	90%	73%



(a) XGB

(b) RF

(c) LR

Figure 3.12 Confusion Matrices (a) XGB, (b) RF, (c) LR

3.1.3 Duplicator Data

The detailed explanation about the dataset is explained in the rest of this section.

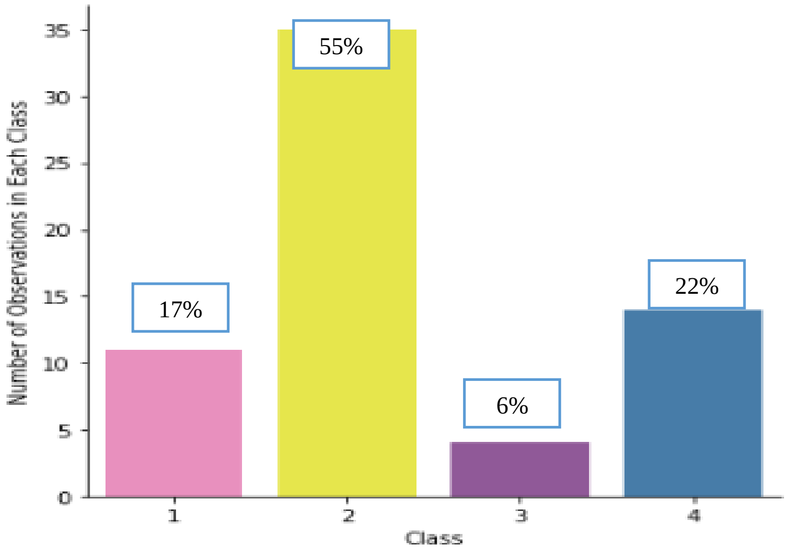


Figure 3.13 Class Distribution of Duplicator Data

55% of the observations belongs to Class II in Figure 3.13. For the other classes, it is 17%, 6%, 22%, respectively. Class III has the least number of observations. Therefore, there is an imbalanced data problem in this dataset, as well. This problem is handled using repeated stratified 3-fold cross validation in Grid Search method. The selected hyperparameters are provided in Table 3.7.

Table 3.7 Surface Defect Data, Grid Search Values for XGB, RF, LR for Duplicator Data

Model	Grid Search Values	Best Parameters	Performance Metric	Best Value
XGB	$subsample$: [0.5, 0.75, 1], $colsample\_bytree$: [0.5, 0.75, 1], $learning\_rate$: [0.01, 0.05, 0.1], max_depth: [2, 4, 6], $gamma$: [0.1, 0.3, 0.5, 1, 2], max_delta_step: [1, 2], $scale\_pos\_weight$: [1, 3, 5]	$subsample$: 1 $colsample\_bytree$: 0.5 $learning\_rate$: 0.05, max_depth: 2, $gamma$: 1, max_delta_step: 1, $scale\_pos\_weight$: 1	Log Loss	1.04
RF	$max\ \_features$: $range$(2,13), $n\_estimators$: $range$(100, 10,000)	$max\ \_features$: 12, $n\_estimators$: 500)	AUC	63%
LR	Default is used	-	-	-

3.1.3.1 XGBoost Tuning

The number of estimators is selected using the results obtained with the GSCV. Different numbers of trees are tried between 100 to 1,000 in the cross-validation to see the performance of XGB in train and test data. The results are evaluated using AUC and Log Loss metrics which are given in Figures 3.14 and 3.15.

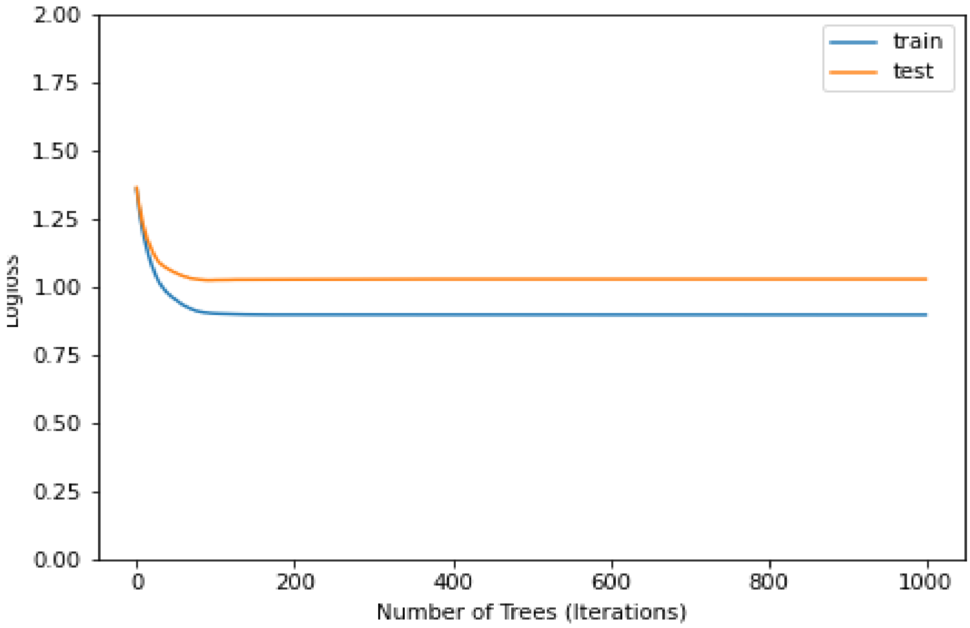


Figure 3.14 XGB, The Number of Trees versus Log Loss for Duplicator Data

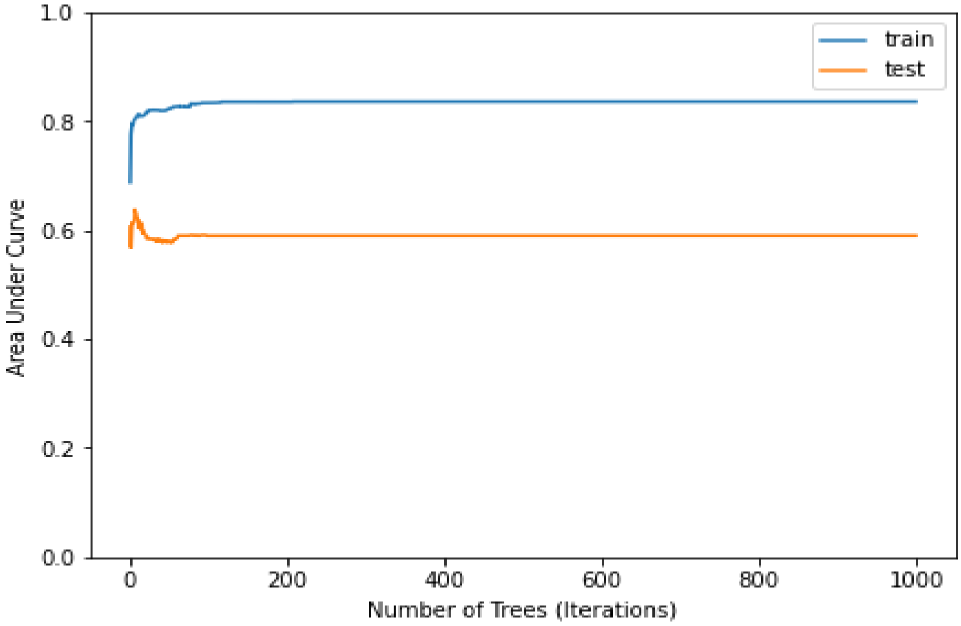


Figure 3.15 XGB, The Number of Trees versus AUC for Duplicator Data

Log Loss decreases first and remains stable for train and test after 200 iterations. For AUC, training scores increase, and test scores decrease between 26-172 iterations. But they are stable after 200 iterations. Thus, the number of trees for XGB model is selected as 200.

3.1.3.2 Random Forest Tuning

The hyperparameters of Random Forest are tuned using OBB error. The obtained results are presented in Figures 3.16 and 3.17.

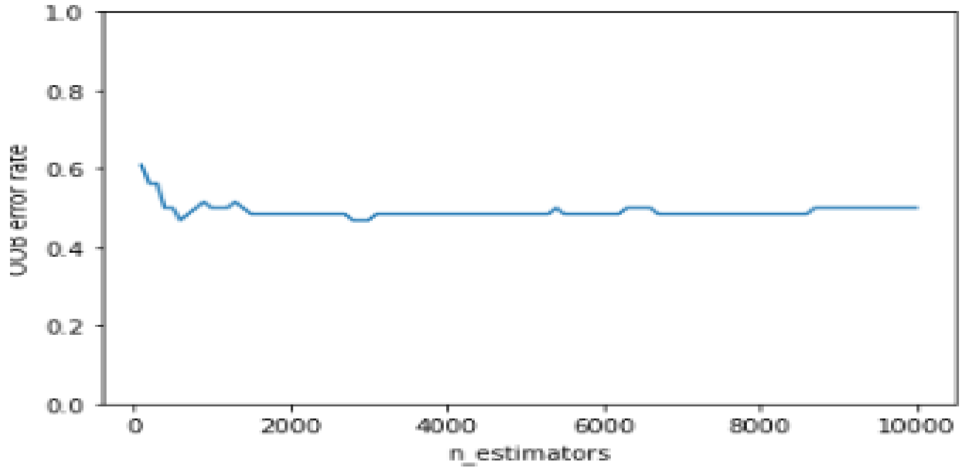


Figure 3.16 RF $n\_estimators$ versus OOB error for Duplicator Data

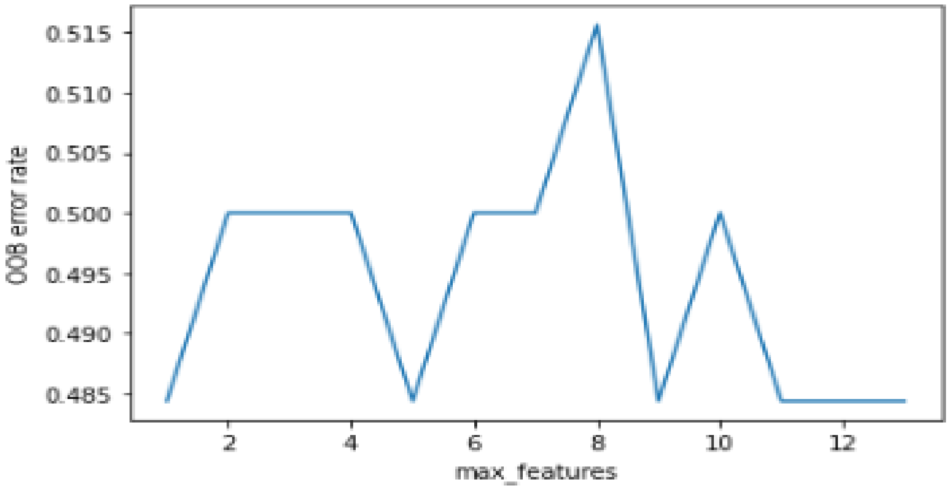


Figure 3.17 RF max $\_features$ versus OOB error for Duplicator Data

OOB error is 0.50 where the number of trees is 500. It is decreasing at first and then it is stabilized around 0.5. So, 500 is selected as the number of trees, as it is validating the GSCV result. It can be seen in Figure 3.18 that OOB error is the lowest where $max\ \_features$ is $5.9, and\ 12$. 12 is selected, since it is parallel to the GSCV result.

3.1.3.3 Performance Results of Models

In Table 3.8, and Table 3.9, Figure 3.18, overall performance results for XGB, RF, and LR are provided.

Table 3.8 Performance Results for XGB, RF, LR for Duplicator Data

Model	**Accuracy**	**F1-score**	**Balanced Accuracy**	**Kappa**	**G-mean**	**AUC**
XGB	**66%**	**66%**	41%	36%	0%	83%
RF	62%	62%	63%	**43%**	54%	**85%**
LR	53%	53%	**64%**	36%	**60%**	**85%**

Table 3.9 Recall, Precision, and Specificity Results for XGB, RF, LR for Duplicator Data

Model	**Recall**				**Precision**				**Specificity**			
	Class I	**Class II**	**Class III**	**Class IV**	**Class I**	**Class II**	**Class III**	**Class IV**	**Class I**	**Class II**	**Class III**	**Class IV**
XGB	0%	91%	0%	71%	0%	67%	0%	62%	100%	45%	100%	88%
RF	18%	71%	100%	64%	50%	78%	25%	75%	96%	76%	80%	94%
LR	45%	46%	100%	64%	31%	80%	25%	75%	79%	86%	80%	94%

XGB shows better performance on accuracy and f-1score. AUC is the same for RF and LR. Also, the LR model has the highest g-mean and kappa.

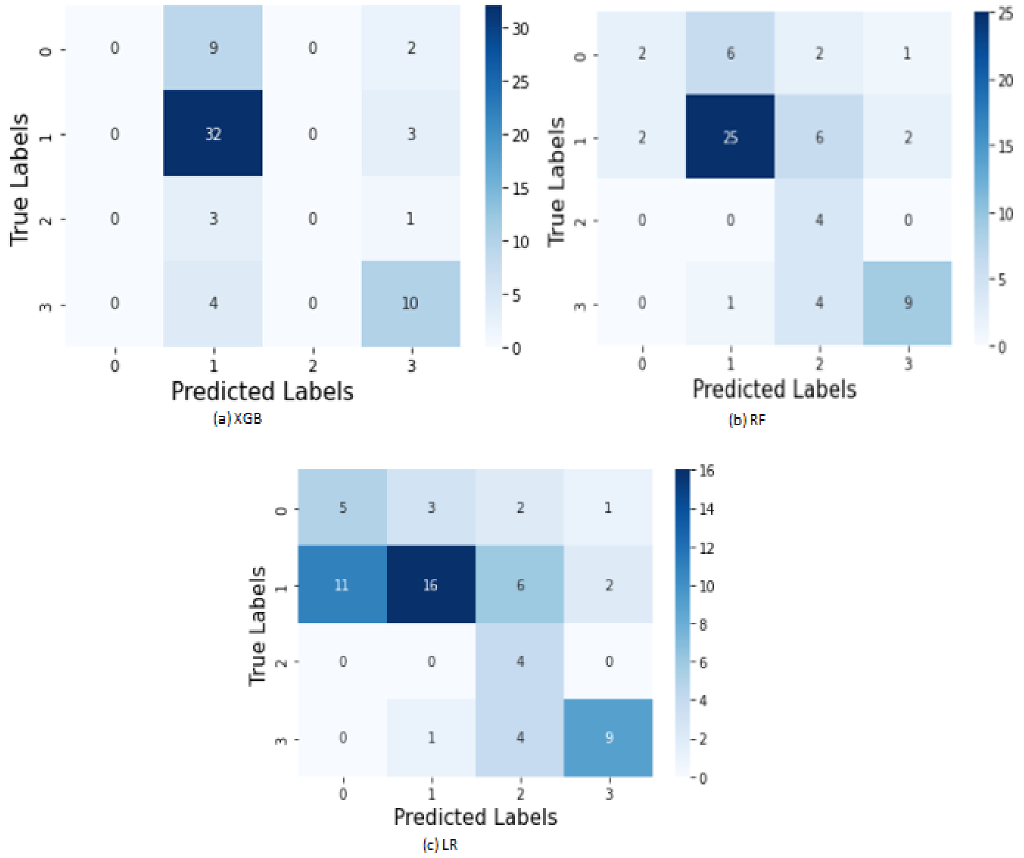


Figure 3.18 Confusion Matrices (a) XGB, (b) RF, (c) LR for Duplicator Data

3.1.4 Foam Molding Data

Due to time limitations, the proposed method could not be applied on this dataset, so the details of foam molding dataset presented in this section, as well. This dataset is analyzed from Jinks (1987), aim is the reducing voids in a urethane-foam product. Then, it is analyzed with different methods (Erdural, 2006; Erişkin et al., 2021; Karabulut, 2013).

There are seven controllable factors and two uncontrollable (noise) factors. They are tabulated in Table 3.10. The response variable has three levels: very good (I), acceptable (II), and needs repair (III). Therefore, it is a smaller-the-better type of problem.

Table 3.10 Controllable and Uncontrollable Factors and Their Levels for Foam Molding Data

Controllable Factors	**Levels**	
	0	1
A. Shot Weight	185	250
B. Mold Temperature	70°F	120°F
C. Foam Block	use	do not use
D. RTV Insert	use	do not use
E. Vent Shell	vented	unvented
F. Spray Wax Viscosity	2:1	4:1
G. Tool Elevation	level	elevated
Uncontrollable Factors	0	1
H. Shift	second	third
I. Shell Quality	good	bad

8 different experiments are conducted, so L_8 orthogonal array is used for this experiment. For each experimental setting, different levels of noise factors are included into the study and 10 repetition is applied. As a result, the number of observations in this dataset is $8 \times 4 \times 10 = 320$. The dataset is given in Table 3.11.

Table 3.11 Experimental Dataset for Foam Molding Case

Exp. No	**Factors**							**H(1)/ I(1)**			**H(0)/ I(1)**			**H(1) / I(0)**			**H(0) / I(0)**		
	A	B	C	D	E	F	G	I	II	III	I	II	III	I	II	III	I	II	III
1	0	0	0	0	0	0	0	3	6	1	6	4	0	1	4	5	0	10	0
2	0	0	0	1	1	1	1	0	3	7	3	4	3	0	6	4	0	7	3
3	0	1	1	0	0	1	1	0	0	10	0	1	9	0	0	10	0	0	10
4	0	1	1	1	1	0	0	0	0	10	0	10	0	0	3	7	0	9	1
5	1	0	1	0	1	0	1	3	5	2	3	7	0	3	5	2	1	6	3
6	1	0	1	1	0	1	0	2	8	0	4	5	1	0	5	5	1	5	4
7	1	1	0	0	1	1	0	2	7	1	2	5	3	2	7	1	1	6	3
8	1	1	0	1	0	0	1	0	4	6	1	7	2	0	4	6	0	3	7

The noise factors cannot be used for the calculations, so the resulting dataset is in Table 3.12.

Table 3.12 The Final Dataset for Foam Molding Case

Exp. No	**Factors**							**Classes**		
	A	B	C	D	E	F	G	I	II	III
1	0	0	0	0	0	0	0	10	24	6
2	0	0	0	1	1	1	1	3	20	17
3	0	1	1	0	0	1	1	0	1	39
4	0	1	1	1	1	0	0	0	22	18
5	1	0	1	0	1	0	1	10	23	7
6	1	0	1	1	0	1	0	7	23	10
7	1	1	0	0	1	1	0	7	25	8
8	1	1	0	1	0	0	1	1	18	21

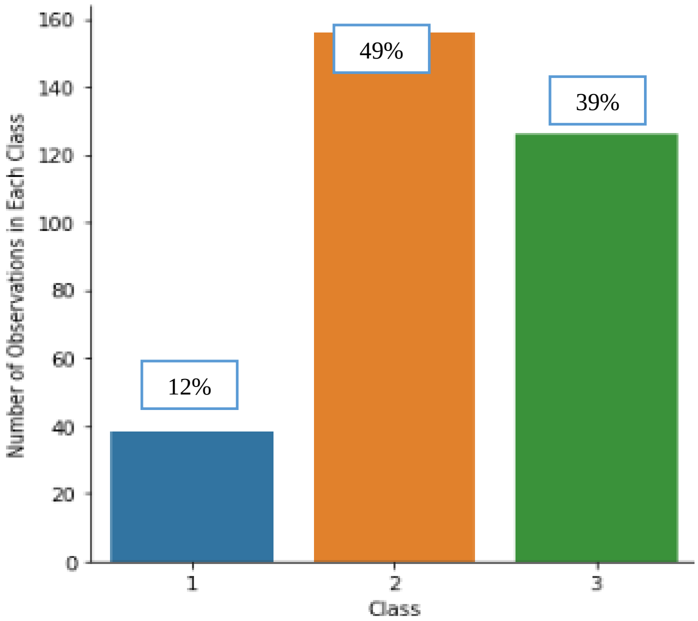


Figure 3.19 Class Distribution of Foam Molding Data

49% of the observations belongs to Class II. 12% of the observations belong to Class I and 39% of them belong to Class III. Class I has less observations than the other two classes, this dataset is slightly imbalanced like the other datasets, as well. This problem is handled using repeated stratified 3-fold cross validation in Grid Search method. The selected hyperparameters are provided in Table 3.13.

Table 3.13 Surface Defect Data, Grid Search Values for XGB, RF, LR for Foam Molding Data

Model	**Grid Search Values**	**Best Parameters**	**Performance Metric**	**Best Value**
XGB	$\mathit{subsample}: [0.5, 0.75, 1]$, $\mathit{colsample\_bytree}: [0.5, 0.75, 1]$, $\mathit{learning\_rate}: [0.01, 0.05, 0.1]$, $\mathit{max\_depth}: [2, 4, 6]$, $\mathit{gamma}: [0.1, 0.3, 0.5, 1, 2]$, $\mathit{max\_delta\_step}: [1, 2]$, $\mathit{scale\_pos\_weight}: [1, 3, 5]$	$\mathit{subsample}: 0.75$ $\mathit{colsample\_bytree}: 0.75$ $\mathit{learning\_rate}: 0.1$, $\mathit{max\_depth}: 4$, $\mathit{gamma}: 0.5$, $\mathit{max\_delta\_step}: 1$, $\mathit{scale\_pos\_weight}: 1$	Log Loss	0.82
RF	$\mathit{max\ \_features}: [2, 3, 4, 5, 6]$, $\mathit{n\_estimators}: \mathit{range}(100, 10{,}000)$	$\mathit{max\ \_features}: 2$, $\mathit{n\_estimators}: 300)$	AUC	68%
LR	Default is used	-	-	-

3.1.4.1 XGBoost Tuning

Using the parameters in Table 3.13, the best value the number of estimators ($n\_estimators$) is selected using Log Loss and AUC. These results are represented for both log loss and AUC metrics in Figures 3.20 and 3.21. Different numbers of trees are tried between 100 to 1,000.

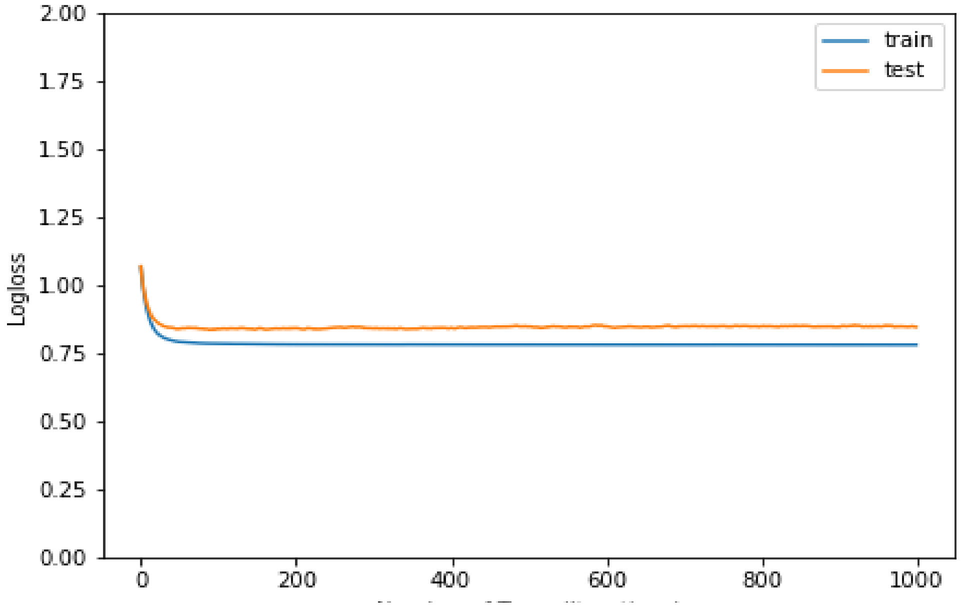


Figure 3.20 XGB, The Number of Trees versus Log Loss for Foam Molding Data

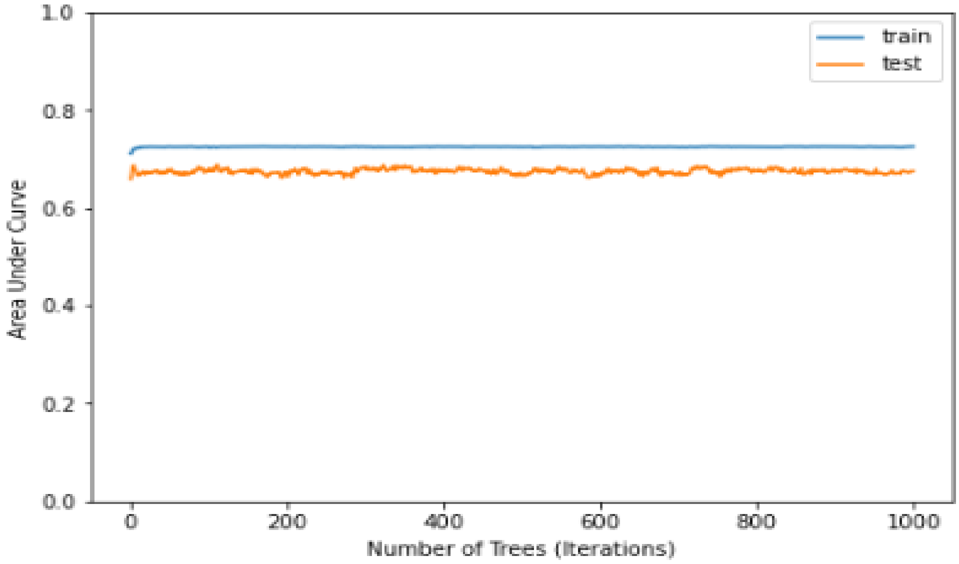


Figure 3.21 XGB, The Number of Trees versus AUC for Foam Molding Data

Log Loss is decreased and stable for train and test after 100 iterations. For AUC, train and test results are close to each other and stable around 0.75. Thus, the number of trees for XGB model is selected as 100.

3.1.4.2 Random Forest Tuning

The hyperparameters of Random Forest are tuned using OBB error. The obtained results are presented in Figures 3.22 and 3.23.

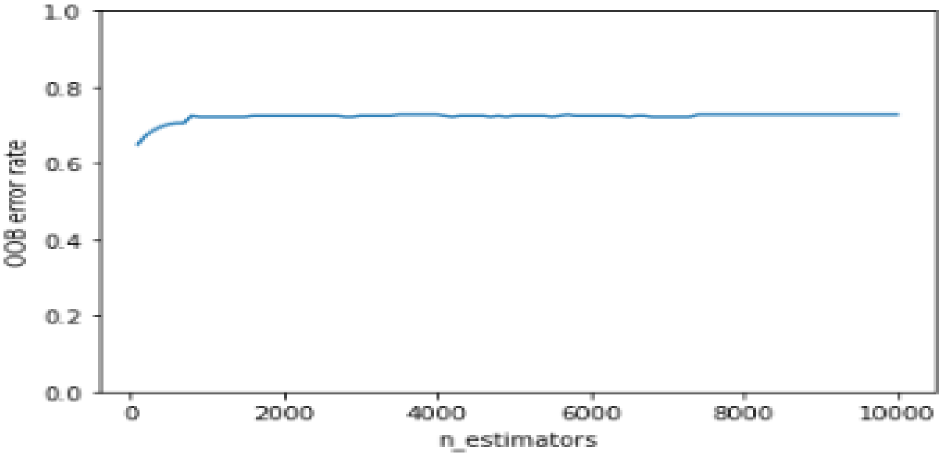


Figure 3.22 RF $n\_estimators$ versus OOB error for Foam Molding Data

OOB error is increasing as the number of trees increases. The number of trees parameter is selected as 300 where OOB error is 0.65.

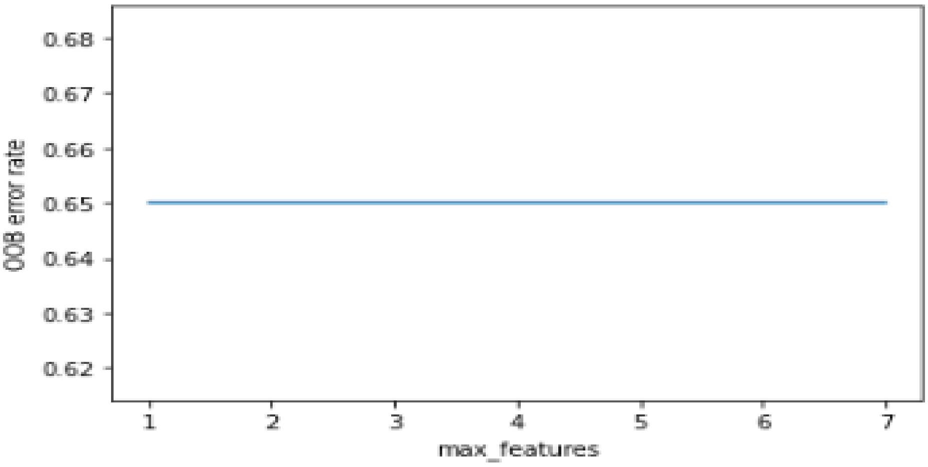


Figure 3.23 RF $max\ \_features$ versus OOB error for Foam Molding Data

OOB error is constant for each maximum feature, so the GSCV result is selected for this parameter.

3.1.4.3 Performance Results of Models

In Table 3.14 and 3.15, Figure 3.24, overall performance results for XGB, RF, and LR are given.

Table 3.14 Performance Results for XGB, RF, LR for Foam Molding Data

Model	**Accuracy**	**F1-score**	**Balanced Accuracy**	**Kappa**	**G-mean**	**AUC**
XGB	**62%**	**62%**	45%	**28%**	0%	**73%**
RF	42%	42%	**55%**	20%	**43%**	72%
LR	40%	40%	**55%**	20%	0%	72%

Table 3.15 Recall, Precision, and Specificity Results for XGB, RF, LR for Foam Molding Data

Model	**Recall**			**Precision**			**Specificity**		
	Class I	**Class II**	**Class III**	**Class I**	**Class II**	**Class III**	**Class I**	**Class II**	**Clas s III**
XGB	0%	88%	48%	0%	57%	75%	100%	37%	90%
RF	89%	14%	61%	21%	55%	64%	55%	89%	78%
LR	89%	0%	75%	21%	0%	59%	55%	100%	66%

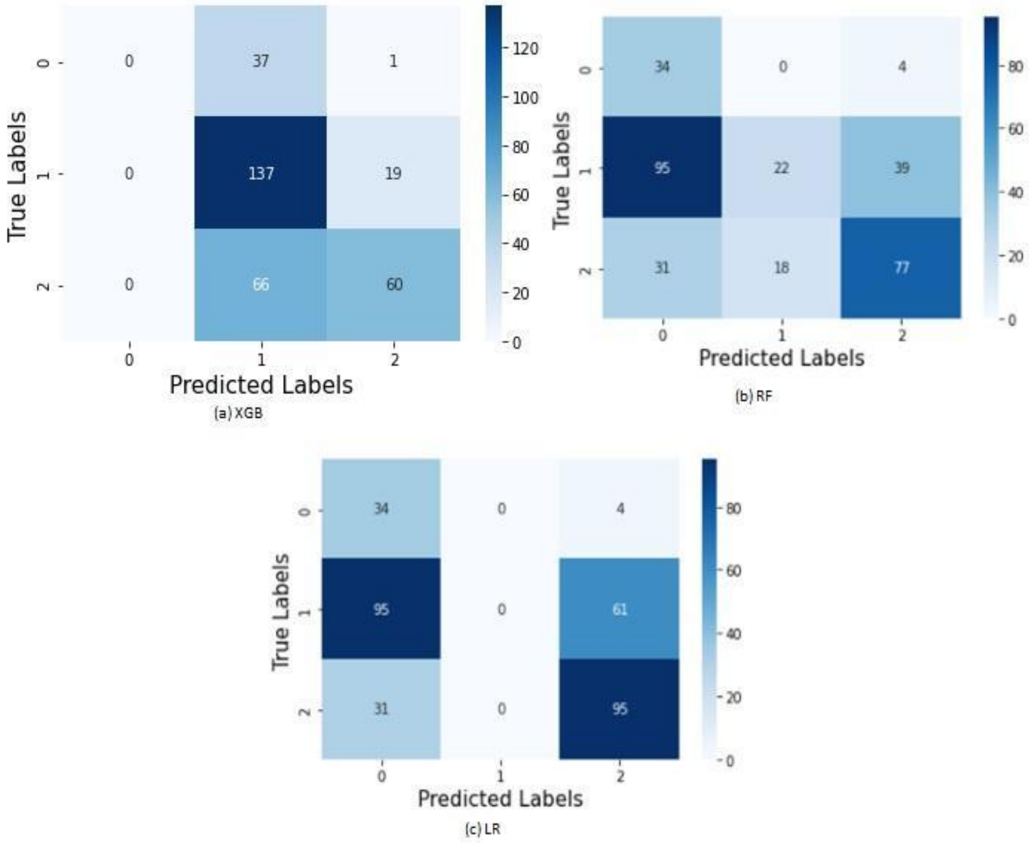


Figure 3.24 Confusion Matrices (a) XGB, (b) RF, (c) LR for Foam Molding Data

XGB shows better performance on accuracy, f-1 score, kappa, and AUC. The RF model has the best performance on g-mean.

3.2 Selecting The Best Method

Stratified 3-fold cross-validation with 3 repetitions is used to validate the models. Recall, precision, and AUC test results are selected as performance measures (see Section 2). The performance results are given in Appendix B.

To check if there is an underfitting or overfitting in our models, the performance is compared with both train and test sets. For all the datasets, it is observed that train and test result are close to each other. Only big differences are in duplicator data.

There is a relatively large differences compared to other results up to 16% on average between train and test data for RF and LR models. However, this difference is not big enough to consider as overfitting. Also, these results are quite close to the XGB results in the same dataset.

This stage is also useful to see the overall performance of our models. In general, cross-validation results are similar to the results that are shown in previous stages. Precision and recall at this stage are calculated as class level and the average of those used to obtain a single metric.

After this step TOPSIS and Multi-MOORA are applied to the average of the test results to rank the models with respect to their performances on recall, precision and AUC criteria. Also, a sensitivity analysis is conducted to TOPSIS to see the effect of each criterion. Two different weighting strategies are adopted: equal weights and entropy weights. For entropy weights, Shannon entropy is calculated to determine disorder degree of criteria (Li et al., 2011). The formula of entropy is given below. Let x_{ij} is the value of alternative i on criteria j.

The benefit criteria are standardized as follows. For cost criteria, it is the same calculation with the min function instead of max.

$$r_{ij} = \frac{x_{ij}}{\max_{j} x_{ij}} \text{ }, i = 1, \dots, m; j = 1, \dots n \tag{3.1}$$

Entropy of the jth criterion is determined by Equation (3.2).

$$H_j = -\frac{\sum_{i=1}^{m} f_{ij} \times ln\ f_{ij}}{ln\, m}, i = 1, \dots, m; j = 1, \dots n \tag{3.2}$$

wherein:

$$f_{ij} = \frac{r_{ij}}{\sum_{i=1}^{m} r_{ij}}, , i = 1, \dots, m; j = 1, \dots n \tag{3.3}$$

Entropy weight of the jth criterion is determined by Equation (3.4)

$$w_j = \frac{1 - H_j}{n - \sum_{j=1}^{n} H_j}, j = 1, \dots n \tag{3.4}$$

The detailed calculations are given in Appendices C to F and the summary of the results are given in Table 3.16.

Table 3.16 Results of TOPSIS and Multi-MOORA

		Method			
		TOPSIS Equal Weight	**TOPSIS Entropy Weights**	**Multi-MOORA Equal Weights**	**Multi-MOORA Entropy Weight**
DataSet	**Alt.**	**Rank**	**Rank**	**Rank**	**Rank**
Surface Defect	RF	2	2	2	2
	LR	3	3	3	3
	XGB	**1**	**1**	**1**	**1**
Duplicator	RF	2	2	2	2
	LR	3	3	3	3
	XGB	**1**	**1**	**1**	**1**
Inkjet Printer	RF	**1**	**1**	**1**	**1**
	LR	3	3	3	3
	XGB	2	2	2	2
Foam Molding	RF	**1**	**1**	**1**	**1**
	LR	2	2	2	2
	XGB	3	3	3	3

For surface defect and duplicator cases, XGBoost ranked as 1st and Random Forest is the 2nd. It is noticed that, for inkjet printer data, Random Forest is ranked as 1st, followed by XGBoost. For these three datasets, Logistic Regression is the least preferable one. However, for the foam molding dataset, Logistic Regression ranked as 2nd and XGBoost model is ranked as last. For all methods, TOPSIS and Multi-

MOORA results are the same with each other. Moreover, for two of the datasets XGBoost ranked as 1st and for the other two datasets Random Forest is the best. It can be said that XGBoost and Random Forest provide better results than Logistic Regression. To see the effect of different weightings, a sensitivity analysis is applied to TOPSIS for both equal and entropy weights. Results are obtained for different weight settings. Similar weighting strategy is adopted from Jiří (2018). Let w_p is the weight of criteria p an w'_p is the new weight of it. Since sum of the weights of all criteria should be 1, this method calculates the other weights as follows.

$$w'_p = w_p + \Delta_p, \;\; p \; \in 1,2,\dots,n \tag{3.5}$$

$$\gamma = \frac{1 - w'_p}{1 - w_p} \tag{3.6}$$

$$w'_j = \gamma \times \; w_j, j \in 1,2,\dots,n, j \neq p \tag{3.7}$$

For example, let $w_1 = w_2 = w_3 = 0.333$, and Δ_1= 0.1. Then,

$$w'_1 = 0.33 + 0.1 = 0.43$$

$$\gamma = \frac{1 - 0.433}{1 - 0.333} = 0.85$$

$$w'_2 = w'_3 = 0.283$$

The weights are changed according to this equation and obtained results are given in Appendix G.

For surface defect data, changing the weights does not change the ranking, XGB ≻ RF ≻ LR. For the duplicator data, for lower weight of AUC ranking is XGB ≻ RF ≻ LR. However, as weight of AUC is increased, the order changes and RF becomes the best alternative, RF ≻ XGB ≻ LR. For duplicator data, it can be said that AUC is an important criterion that affects the ranking. Similar situation is observed in inkjet data. For lower AUC ranking is RF $\succ$ XGB $\succ$ LR, but order change for higher AUC as XGB $\succ$ RF $\succ$ LR. XGB is the best alternative. Finally, for foam molding data for higher weights of recall, ranking is RF $\succ$ LR $\succ$ XGB. However, for lower level of recall, ranking changes and XGB outperform the LR, RF $\succ$ XGB $\succ$ LR.

As a conclusion, for these datasets, precision is robust to weight changes, ranking is not affected by it. For the higher values of AUC, the best method changes. Also, for only one dataset it is seen that as recall becomes less important, XGB ranks before LR. Among all these results, XGB and RF are selected as the best and LR performs worse than those methods and is ranked as last. The performance of Random Forest and XGBoost seems close to each other; both perform well for these experimental datasets. Any user can use either Random Forest or XGBoost. At this point, Random Forest is selected as the data mining algorithm to be used for the rest of the study because it is easier to implement compared to XGBoost.

3.3 Steps of The Proposed Method

Based on the literature review and analyses performed, it is decided to use RF to predict class probabilities and median and COV as measures of location and dispersion respectively. Based on these decisions, steps of the proposed method are illustrated in Figure 3.25.

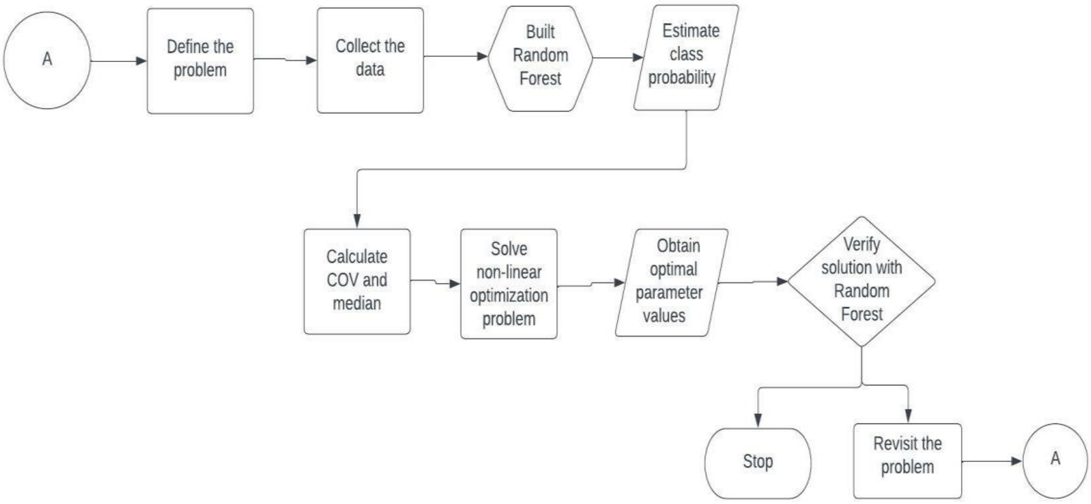

A
Define the problem
Collect the data
Built Random Forest
Estimate class probability
Calculate COV and median
Solve non-linear optimization problem
Obtain optimal parameter values
Verify solution with Random Forest
Stop
Revisit the problem
A

Figure 3.25 Flowchart of The Proposed Method

Step 1: Collect the experimental data

Montgomery (2020) provides a procedure for designing experiments. The steps of the procedure are:

1. Statement of problem
2. Selection of factors and the levels
3. Selection of response variable
4. Choice of experimental design
5. Performing the experiment and analyzing the data

The problem should be clearly defined in pre-experiment stage. This step is important for understanding the problem and developing the objectives. Afterwards, the factors and the ranges of each factor should be decided. Then, the experimenter should select the response variable. Here, one important thing is, the response variable should provide useful information about the experiment. The steps up to this point is, part of pre-experimental planning.

The choice of experimental design step involves selection of run order for the experiment, the number of replications, and decision of whether any randomization restrictions are used. An experimenter can choose designs such as: Factorial design, Fractional Design, Central Composite Design etc. Afterward, experiments are performed and monitored to check if everything is working according to the plan, and finally results are analyzed using appropriate statistical methods. Taguchi used orthogonal designs where an orthogonal array involving x (controllable factor settings, inner array) is crossed with an orthogonal array involving z (noise factor settings, outer array). In this type of design every level of a factor occurs with every level of other factors the same number of times (Logothetis, 1992; Robinson et al., 2004).

An example of the experimental dataset that will be used when the response is categorical is given in Table 3.17. Let there are N experiments with r replications, therefore resulting dataset will contain $N \times r$ observations. $x_1, \dots, x_p$ be the controllable factors (design factors) of product or process and Y is the ordinal response with K categories. $y_1, \dots, y_{(N\times r)}$ is the output of the Random Forest model, that is, y_1 is K if the 1st observation is classified as Class K.

Table 3.17 An Example Dataset for The Proposed Method

Obs. No.	**Exp. No.**	**Rep. No.**	**Design Parameters**				**Response of The Experiment**
			x_1	x_2	$\cdots$	x_p	Y_k
1	1	1	x_{11}	x_{12}	$\cdots$	x_{1p}	y_1
2	1	2	x_{21}	x_{22}	$\cdots$	x_{2p}	y_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\cdots$		$\vdots$
r	1	r	x_{r1}	x_{r2}	$\cdots$	x_{rp}	y_r
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$		$\vdots$
$(N-1)r+1$	N	1	$x_{(N-1)r+11}$	$x_{(N-1)r+12}$	$\cdots$	$x_{(N-1)r+1p}$	$y_{((N-1)r+1)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$N\times r$	N	r	$x_{(N\times r)1}$	$x_{(N\times r)2}$	$\cdots$	$x_{(N\times r)p}$	$y_{(N\times r)}$

Step 2: Build Random Forest model

Random Forest is used to obtain the probability of each class. Therefore, fitting a Random Forest model is an important step for the proposed method. The two important hyperparameters of Random Forest are number of trees and number of attributes used in each node (Breiman, 2001; Sagi & Rokach, 2018). In the proposed method, the performance is compared for $100\ to\ 10{,}000$ trees and $2\ to\ p$ features. To tune these parameters, two different methods are applied.

- Searching best hyperparameters with Grid Search Cross Validation (GSCV)
- Tuning these parameters based on OBB Error

Random Forest is available for various software packages. The Scikit-learn library (Pedregosa et al., 2011) in Python 3.10.0 (Python, 2021) is used in this study. Grid Search Cross Validation tries to find the best parameter setting based on a predefined performance metric. The choice of this performance metric can be done according to the type of data. If there is an imbalanced data problem in the dataset, AUC can provide a better result (Hossin & Sulaiman, 2015).

For the OOB error tuning, the Random Forest model is fitted using different values for the number of trees and maximum attributes used to split a node and then OBB error rates are monitored. The point providing the least OBB error can be used as hyperparameter. However, sometimes the result for GSCV and OOB error can be in conflict since GSCV tries to maximize a certain performance metric and OBB error rates method tries to minimize the error. In such cases, using parameters obtained according to the OBB error can be meaningful because OOB error provides unbiased estimation compared to cross-validation (Breiman, 2001).

After obtaining the best parameter setting, the model is fitted, and class probabilities are obtained. The Random Forest model is obtained with majority voting. Therefore, class probabilities are calculated as the proportion of number of trees which classify experiment i as j, to the number of trees in the forest.

$$\widehat{p_{ij}} = \frac{Number\ of\ trees\ which\ classify\ the\ experiement\ i\ as\ j}{Number\ of\ trees\ in\ the\ forest} \qquad (3.8)$$

where $0 \leq \widehat{p_{ıȷ}} \leq 1, j = 1, . . , K, i = 1, \ldots, N$

Step 3: Calculate COV and median

The estimated probability obtained from Random Forest is used to calculate COV and median at this step. The reason of using COV and median to analyze categorical data is explained detailly in Section 2. These metrics are calculated for each experiment point as follows.

$$\widehat{COV_ı} = 1 - \left(\frac{\sum_{j=1}^{K-1} \left|2F_j - 1\right|^2}{K-1}\right)^{\frac{1}{2}} i = 1, \ldots, N \tag{3.9}$$

where K is the number of classes (categories).

$$\widehat{Medıan_ı} = (q| \sum_{j=1}^{q-1} \hat{P}(Y = j \mid x_i) < 0.5, \sum_{j=q}^{K} \hat{P}(Y = j \mid x_i) \geq 0.5), \ i = 1, \ldots., N \tag{3.10}$$

N: Number of experimental data points

$\hat{P}(Y = j \mid x_i)$ = Estimated probability of experiment i belongs to class j

$\widehat{COV_ı}$ = Estimated COV for experimental trial i, $i = 1, \ldots, N$

$\widehat{Medıan_ı}$ = Estimated median of experimental trial i, $i = 1, \ldots, N$

Step 4: Find optimal design parameters with minimum COV and target quality level

In the previous step, COV and median are calculated for each experimental data point. In order to obtain the optimal parameter design, empirical models that relate COV and median with design parameters are built.

In order to build the model that estimate COV for any design parameter setting, the least square regression is used. Since, median is a categorical measure, it is preferred to build the empirical model of median with Ordinal Logistic Regression. As mentioned in Section 2.3.8 different link functions can be used in Ordinal Logistic Regression. When the gompit link function is used, the estimated probability that the response category (median class) is less than or equal to j, at control factor level combination $\boldsymbol{x}$ is obtained using Equation (3.12). Using the fitted model, estimated probability that the median class is j is obtained by Equations (3.13) and (3.14). Then, median class is selected as the one that has the highest probability as shown in Equation (3.15).

$$\widehat{COV} = \hat{\beta}_0 + \sum_{i=1}^{p} \hat{\beta}_i x_i \tag{3.11}$$

$$\hat{P}\big(\tilde{Y}(\boldsymbol{x}) \le j\big) = 1 - exp\left(- exp\left(\widehat{\gamma_j} + \sum_{i=1}^{p} \hat{\theta}_i x_i\right)\right) \tag{3.12}$$

$$\hat{P}\big(\tilde{Y}(\boldsymbol{x}) = j\big) = \hat{P}\big(\tilde{Y}(\boldsymbol{x}) \le j\big) - \hat{P}\big(\tilde{Y}(\boldsymbol{x}) \le j-1\big), j = 2,3,\ldots,K \tag{3.13}$$

$$\hat{P}\big(\tilde{Y}(\boldsymbol{x}) = 1\big) = \hat{P}\big(\tilde{Y}(\boldsymbol{x}) \le 1\big) \tag{3.14}$$

$$\widehat{Median} = argmax(\hat{P}\big(\tilde{Y}(\boldsymbol{x}) = 1\big), \hat{P}\big(\tilde{Y}(\boldsymbol{x}) = 2\big), \ldots, \hat{P}\big(\tilde{Y}(\boldsymbol{x}) = K\big)) \tag{3.15}$$

where $\widehat{\gamma_j}$ is the intercept of class $j, j = 1, \ldots K$, $\boldsymbol{x}$ is the set of controllable factors, $\hat{P}\big(\tilde{Y}(\boldsymbol{x}) \le j\big)$ denotes the estimated probability that the median class is less than or equal to j and $\widehat{\boldsymbol{\beta}}$ and $\widehat{\boldsymbol{\theta}}$ are the set of regression coefficients.

Once the COV and median are modeled as functions of the design parameters, a non-linear optimization model is used to find optimal levels of the design parameters with minimum COV and desired median value. The mathematical model for different problem types is given in Table 3.18.

Table 3.18 Mathematical models for the different types of problems

Type of Problem	Mathematical Model
Smaller-the-Better	$Min\ \widehat{Median}\ = \boldsymbol{f}(x_1, x_2, \dots, x_p)$ $Min\ \widehat{COV} = \boldsymbol{g}(x_1, x_2, \dots, x_p)$ $s.t.$ $l_i \leq x_i \leq u_i\ , i = 1, 2, \dots, p$ $\widehat{Median}\ \geq 0$ $0 \leq \widehat{COV} \leq 1$
Larger-the-Better	$Max\ \widehat{Median} = \boldsymbol{f}(x_1, x_2, \dots, x_p)$ $Min\ \ \widehat{COV} = \boldsymbol{g}(x_1, x_2, \dots, x_p)$ $s.t.$ $l_i \leq x_i \leq u_i\ , i = 1, 2, \dots, p$ $\widehat{Median}\ \geq 0$ $0 \leq\ \widehat{COV} \leq 1$

To combine the two objectives into a single objective, rather than minimizing or maximizing the median, the bias between optimal median and target level is minimized (Ding et al., 2007; D. Lee et al., 2016; D. H. Lee & Kim, 2012). Weighted mean square error is used to aggregate these two different objective functions (Ding et al., 2007). Equal importances are given to these objectives. The short form of our mathematical model is as follows.

$$Min\, w_{Median} \times \left(\widehat{Median} - T\right)^2 + w_{COV} \times \left(\widehat{COV}\right)^2$$

$s.t.$

$$\widehat{COV} = \widehat{\beta_0} + \sum_{i=1}^{p} \widehat{\beta}_i x_i \tag{3.16}$$

$$\hat{P}\big(\tilde{Y}(\boldsymbol{x}) \le j\big) = 1 - exp\ \left(- exp\ \left(\hat{\gamma}_j + \textstyle\sum_{i=1}^{p} \hat{\theta}_i x_i\right)\right) \tag{3.17}$$

$$\hat{P}\big(\tilde{Y}(\boldsymbol{x}) = j\big) = \hat{P}\big(\tilde{Y}(\boldsymbol{x}) \le j\big) - \hat{P}\big(\tilde{Y}(\boldsymbol{x}) \le j-1\big), j = 2,3,\ldots,K \tag{3.18}$$

$$\hat{P}\big(\tilde{Y}(\boldsymbol{x}) = 1\big) = \hat{P}\big(\tilde{Y}(\boldsymbol{x}) \le 1\big) \tag{3.19}$$

$$\widehat{Median} = argmax\ \left(\hat{P}\big(\tilde{Y} = 1\big), \hat{P}\big(\tilde{Y} = 2\big), \ldots, \hat{P}(\tilde{Y} = K)\right) \tag{3.20}$$

$$l_i \le x_i \le u_i\, , i = 1, 2, \ldots, p \tag{3.21}$$

$$0 \le\ \widehat{COV} \le 1 \tag{3.22}$$

$$\widehat{Median} \ge 0 \tag{3.23}$$

where T is the target quality level and $\widehat{\beta_i}$ is the estimated coefficients of design parameters. In order to convert *argmax* function to constraints, binary decision variables are included to the model. Therefore, the final model would be like:

Decision variables:

$\hat{P}\big(\widetilde{Y}(\boldsymbol{x}) = j\big) = The\ estimated\ probability\ that\ median\ category\ is\ j\ at\ \boldsymbol{x}, \forall j = 1,2,..,K$

$\hat{P}(\widetilde{Y} = max)\ = maximum\ probability\ of\ all\ \hat{P}(\widetilde{Y} = j)'s\ .$

$$b_j = \begin{cases} 1, if\ \hat{P}\big(\widetilde{Y} = j\big)\ is\ the\ maximum\ of\ all\ \hat{P}\big(\widetilde{Y} = j\big)'s\ \ \forall j = 1,2,..,K \\ \qquad 0,\ otherwise \end{cases}$$

$$Z_j = \begin{cases} j, if \ \hat{P}(\tilde{Y} = j) \ is \ the \ maximum \ of \ all \ \hat{P}(\tilde{Y} = j)' s \ , \forall j = 1,2,..,K \\ 0, otherwise \end{cases}$$

Mathematical Model:

$$Min \ w_{Median} \times \left(\widehat{Med\imath an} - T\right)^2 + w_{COV} \times \left(\ \widehat{COV}\right)^2$$

$s.t.$

$$\widehat{COV} = \widehat{\beta_0} + \sum_{i=1}^{p} \widehat{\beta_\imath} x_i, i = 1,2,\dots,p \tag{3.21}$$

$$\hat{P}(\tilde{Y}(\boldsymbol{x}) = 1) = 1 - exp\ \left(- \ exp\ \left(\widehat{\alpha_1} + \sum_{i=1}^{p} \hat{x}_i \hat{\beta}_i\right)\ \right) \tag{3.22}$$

$$\hat{P}(\tilde{Y}(\boldsymbol{x}) = j) = 1 - exp\ \left(- \ exp\ \left(\widehat{\alpha_j} + \sum_{i=1}^{p} \hat{x}_i \widehat{\beta_\imath}\right)\ \right) - \sum_{k=1}^{j-1} \hat{P}(\tilde{Y}(\boldsymbol{x}) = k) \ \forall j, = 2, \dots K \tag{3.23}$$

$$\hat{P}(\ \tilde{Y} = max) \ \leq \ \hat{P}(\tilde{Y} = j) + (1 - b_j) \ \forall j = 1,2,..,K \tag{3.24}$$

$$\hat{P}(\ \tilde{Y} = max) \geq \ \hat{P}(\tilde{Y} = j) \ \forall j = 1,2,..,K \tag{3.25}$$

$$\sum_{j=1}^{K} b_j = 1 \ \ \forall j = 1,2,..,K \tag{3.26}$$

$$Z_j = j \times b_j \ \forall j = 1,2,..,K \tag{3.27}$$

$$\widehat{Med\imath an} = \ \sum_{j=1}^{K} Z_j \tag{3.28}$$

$$0 \leq \hat{P}(\ \tilde{Y} = max) \leq 1 \tag{3.29}$$

$$\widehat{Med\imath an} \ \in \{1,2, \dots K\} \tag{3.30}$$

$$0 \leq \ \widehat{COV} \leq 1 \tag{3.31}$$

$$b_j \in \{0, 1\} \ \forall j = 1,2,..,K \tag{3.32}$$

The maximum probability among all probabilities is selected using constraints 3.26 and 3.27, and the index of it specified with the help of binary decision variable b_j. Then, optimal median is selected using constraints 3.29 and 3.30. Note that, decision variable Z_j can be 0 or the j. Therefore, sum of all Z_j would be equal to the optimal median value. This non-linear optimization model is solved to obtain an optimal solution with appropriate software. In this study MATLAB / BARON solver is used (MathWorks, 2018).

Step 5: Verification of results with Random Forest model

After obtaining the optimal design parameters, COV, and median, these results are tested with the Random Forest model built in step 3. The optimal parameters are used to predict class probabilities. For the new estimated probabilities, COV and median are calculated again using Equations (3.9) and (3.10) respectively to double check optimization results.

If the results of non-linear model and Random Forest do not parallel each other, the following steps should be revisited.

1. Check whether Random Forest fits data well. Using cross validation, the user can compare performance on both train and test data. If there is a large gap between these values, there might be an overfitting problem. In that case, Random Forest should be built again with a more appropriate hyperparameter setting.
2. Check linear regression and Ordinal Logistic Regression models used for estimating COV and median.
3. Use different starting points for the non-linear optimization model.

3.4 Case Study I: Surface Defect

In this section, the proposed method is applied to the surface defect dataset. This case study examines defects of deposition of a polysilicon process which is used in producing very large scale integrated (VLSI) circuits. Phadke (1989) mentions the importance of this process as a large number of wafers are not used due to the excessive defects.

Step 1: Collect the experimental data

To minimize defects, Phadke (1989) introduces six controllable factors at three levels listed in Table 3.19. These are: Deposition temperature (℃) (A), Deposition pressure ($mtorr$) (B), Nitrogen flow ($sccm$) (C), Silane flow ($sccm$) (D), Settling time (min) (E), and Cleaning method (F). Data is collected using Taguchi Robust Parameter Design.

Table 3.19 Controllable Factors and Their Levels (Source: (Phadke, 1989))

Factors	**Levels**		
	1	2	3
A. Deposition temperature (℃)	$T_0 - 25$	T_0	$T_0 + 25$
B. Deposition pressure ($mtorr$)	$P_0 - 200$	P_0	$P_0 + 200$
C. Nitrogen flow ($sccm$)	N_0	$N_0 - 150$	$N_0 - 75$
D. Silane flow ($sccm$)	$S_0 - 100$	$S_0 - 50$	S_0
E. Settling time (min)	t_0	$t_0 + 8$	$t_0 + 16$
F. Cleaning method	None	CM_2	CM_3

Factor C does not specify an order, so the order of it is changed by Karabulut (2013). The rearranged levels of factor C (C') are in Table 3.20.

Table 3.20 Rearranged Levels of Factor C

Levels	**C**	**C'**
N_0	1	3
$N_0 - 75$	3	2

Table 3.20 (cont'd) Rearranged Levels of Factor C

$N_0 - 150$	2	1

Since the objective is minimizing the number of defects, this is a smaller-the-better type of problem. Phadke (1989) uses L_{18} orthogonal array for this experiment, with 9 repetitions. Total number of observations in the dataset is $18 \times 9 = 162$. The number of defects is counted on the different areas on the wafer and recorded as an ordered categorical variable. These five categories are: no surface defect, very few defects, some defects, many defects, and too many defects as provided in Table 3.21. In this case, a desirable quality level is obtained with the first class.

Table 3.21 Number of Defects for Each Class

Classes	**Number of Surface Defects**
I	0-3 defects
II	4-30 defects
III	31-300 defects
IV	300-1000 defects
V	1001 or more defects

The resulting dataset is tabulated as given in Table 3.22:

Table 3.22 Experimental Dataset for Surface Defects Case

Exp. No.	**Factors**							**Number of Observations by Classes**				
	A	B	C'	C	D	E	F	I	II	III	IV	V

Table 3.22 (cont'd) Experimental Dataset for Surface Defects Case

1	1	1	3	1	1	1	1	9	0	0	0	0
2	1	2	1	2	2	2	2	5	2	2	0	0
3	1	3	2	3	3	3	3	1	0	6	2	0
4	2	1	3	1	2	2	3	0	8	1	0	0
5	2	2	1	2	3	3	1	0	1	0	4	4
6	2	3	2	3	1	1	2	1	0	4	1	3
7	3	1	1	2	1	3	3	0	1	1	4	3
8	3	2	2	3	2	1	1	3	0	2	1	3
9	3	3	3	1	3	2	2	0	0	0	4	5
10	1	1	2	3	3	2	1	9	0	0	0	0
11	1	2	3	1	1	3	2	8	1	0	0	0
12	1	3	1	2	2	1	3	2	3	3	0	1
13	2	1	1	2	3	1	2	4	2	2	1	0
14	2	2	2	3	1	2	3	2	3	4	0	0
15	2	3	3	1	2	3	1	0	1	1	1	6
16	3	1	2	3	2	3	2	3	4	2	0	0
17	3	2	3	1	3	1	3	2	1	0	2	4
18	3	3	1	2	1	2	1	0	0	0	2	7

Step 2 and 3: Build Random Forest model, Calculate COV and median

After the hyperparameter tuning stage explained in Section 3.1.1.2, the best parameter setting is used to fit the model. Class probabilities at each experimental trial are estimated using the resulting RF model. Then, these probabilities are used to calculate COV and median values by Equations (3.9) and (3.10) respectively. Estimated probabilities, cumulative probabilities, and calculated COV and median values are given Table 3.23.

Table 3.23 Estimated Class Probabilities, COV and Median for Surface Defect Case

Exp. No.	$\hat{P}(Y = I)$	$\hat{P}(Y = II)$	$\hat{P}(Y = III)$	$\hat{P}(Y = IV)$	$\hat{P}(Y = V)$	$\widehat{F_Y}(I)$	$\widehat{F_Y}(II)$	$\widehat{F_Y}(III)$	$\widehat{F_Y}(IV)$	$\widehat{F_Y}(V)$	$\widehat{COV}$	$\widehat{Median}$
1	1	0	0	0	0	1	1	1	1	1	0	1
2	0.4282	0.2928	0.2791	0	0	0.4282	0.7209	1	1	1	0.2557	2
3	0.0687	0	0.6667	0.2641	0.0005	0.0687	0.0687	0.7354	0.9995	1	0.1772	3
4	0	0.8936	0.1064	0	0	0	0.8936	1	1	1	0.0487	2
5	0	0.1175	0	0.5335	0.3490	0	0.1175	0.1175	0.6510	1	0.2481	4
6	0.0737	0	0.4868	0.1494	0.2902	0.0737	0.0737	0.5604	0.7098	1	0.3587	3
7	0	0.1090	0.1085	0.5277	0.2547	0	0.1091	0.2176	0.7453	1	0.2633	4
8	0.2516	0	0.2622	0.1702	0.3160	0.2516	0.2516	0.5138	0.6840	1	0.6032	3
9	0	0	0	0.5539	0.4461	0	0	0	0.5539	1	0.1323	4

Table 3.23 (cont'd) Estimated Class Probabilities, COV and Median for Surface Defect Case

10	0.9998	0.0002	0	0	0	0.9998	1	1	1	1	0.0001	1
11	0.8336	0.1664	0	0	0	0.8336	1	1	1	1	0.0719	1
12	0.1487	0.3775	0.3717	0.0003	0.1019	0.1487	0.5261	0.8979	0.8981	1	0.3360	2
13	0.3212	0.2635	0.2542	0.1610	0	0.3212	0.5848	0.8390	1	1	0.3644	2
14	0.1469	0.3705	0.4827	0	0	0.1469	0.5173	1	1	1	0.2094	2
15	0	0.1200	0.1190	0.1567	0.6042	0	0.1200	0.2390	0.3958	1	0.3120	5
16	0.2235	0.5296	0.2469	0	0	0.2235	0.7531	1	1	1	0.1997	2
17	0.1542	0.1265	0	0.3131	0.4062	0.1542	0.2807	0.2807	0.5938	1	0.5262	4
18	0	0	0	0.3047	0.6953	0	0	0	0.3047	1	0.1122	5

Among the calculated COV and median values, experiment 1st, $A_1B_1C'_1D_1E_1F_1$ seems as a good alternative. The COV value is 0 and the median is at the most desirable level, 1. On the other hand, experiment 10th ($A_1B_1C'_3D_3E_2F_1$) can be a good alternative to 1st. Its COV value is very close to 0 and median is 1. These are the best parameter designs among the ones tested before, but the best parameter design can be also in the non-tested designs, as well. To find those designs, a non-linear optimization problem is solved.

Step 4: Find optimal design parameters with minimum COV and target quality level

In Table 3.19, factors A, B, C', D and E are given as continuous attributes, and factor F is given as nominal. Their original data type is preserved while building the regression models.

For COV, a linear regression is fitted using values in Table 3.23. First, both main factors and their two-way interactions are considered as the potential independent variables of the regression model. Using the stepwise regression option in Minitab 20 (2020) and taking significance level to enter or remove a variable as 0.2, an initial model is obtained. Then, to find the most significant terms in the model, factors are added and removed from the model according to their significance by considering a significance level of 0.05. Factor F does not take place in the final model since it is not found as significant. In the final model, R^2 and adjusted-R^2 are 94.66% and 90.93%, respectively, which are at satisfactory level. Also, all p-values are less than 0.05, which means all terms are significant. Then an Ordinal Logistic Regression model that relate median with the controllable factors is fitted. Logit, gompit, and probit link functions are tried. However, only gompit link function provide reasonable results. For fitting the Ordinal Logistic Regression, first all the main factors are added to the model and least significant ones are removed. Then, two-way interactions of the significant factors are added. All p-values obtained with the final model are less than 0.05, therefore, all terms are significant. The performances of linear regression and logistic regression models are given in Appendix H. The obtained mathematical functions are as follows:

$$\widehat{COV} = 0.2040 + 0.0956 \times A - 0.0770 \times E - 0.1648 \times B^2 + 0.2103 \times E^2 + 0.0954 \times A \times C - 0.0980 \times A \times D + 0.0504 \times C \times E$$

$$\hat{P}(\tilde{Y} \leq 1) = 1 - exp\,(- exp\,\,(7.90192 - 2.61839 \times A - 2.08751 \times B - 1.16754 \times E)\,)$$

$$\hat{P}(\tilde{Y} \leq 2) = 1 - exp\,(- exp\,\,(11.5900 - 2.61839 \times A - 2.08751 \times B - 1.16754 \times E)\,)$$

$$\hat{P}(\tilde{Y} \leq 3) = 1 - exp\,(- exp\,\,(12.9148 - 2.61839 \times A - 2.08751 \times B - 1.16754 \times E)\,)$$

$$\hat{P}(\tilde{Y} \leq 4) = 1 - exp\,(- exp\,\,(14.9553 - 2.61839 \times A - 2.08751 \times B - 1.16754 \times E)\,)$$

Factor F is not found as significant in terms of both linear regression and ordinal logistic regression models. To find the best parameter design, a non-linear optimization problem is solved by using Baron Solver (2018). There are two objectives:

1. Reducing variability, therefore COV is minimized.
2. Reducing the difference between the response and the target value.

Weighted Mean Square Error (WMSE) combines these two objectives in a single equation. Target quality level is 1, so bias is obtained by subtracting 1 from the estimated median. Equal importances are given to COV and median, $w_{cov} = w_{median} = 0.5$.

$$Min\;\; 0.5 \times \left(\widehat{COV}\right)^2 + 0.5 \times \left(\widehat{Median} - 1\right)^2$$

s.t.

$$\widehat{COV} = 0.2040 + 0.0956 \times A - 0.0770 \times E - 0.1648 \times B^2 + 0.2103 \times E^2 + 0.0954 \times A \times C - 0.0980 \times A \times D + 0.0504 \times C \times E$$

$$\hat{P}(\tilde{Y} = 1) = 1 - exp\,(- exp\ (7.90192 - 2.61839 \times A - 2.08751 \times B - 1.16754 \times E)\,)$$

$$\hat{P}\big(\tilde{Y} = 2\big) = 1 - exp\,(- exp\ (11.5900 - 2.61839 \times A - 2.08751 \times B - 1.16754 \times E)\,) - \hat{P}\big(\tilde{Y} \leq 1\big)$$

$$\hat{P}\big(\tilde{Y} = 3\big) = 1 - exp\,(- exp\ (12.9148 - 2.61839 \times A - 2.08751 \times B - 1.16754 \times E)\,) - \hat{P}\big(\tilde{Y} \leq 2\big)$$

$$\hat{P}\big(\tilde{Y} = 4\big) = 1 - exp\,(- exp\ (14.9553 - 2.61839 \times A - 2.08751 \times B - 1.16754 \times E)\,) - \hat{P}\big(\tilde{Y} \leq 3\big)$$

$$\sum_{i=1}^{5} \hat{P}(Y = i) = 1$$

$$\hat{P}\big(\ \tilde{Y} = max\big) \leq \ \hat{P}\big(\tilde{Y} = i\big) + (1 - x_i),\ \forall i = 1,2,..,5$$

$$\hat{P}\big(\ \tilde{Y} = max\big) \geq \ \hat{P}\big(\tilde{Y} = i\big),\ \forall i = 1,2,..,5$$

$$\sum_{i=1}^{5} x_i = 1,\ \forall i = 1,2,..,5$$

$$Z_i = i \times x_i\ \forall i = 1,2,..,5$$

$$\widehat{Med\imath an} = \ \sum_{i=1}^{5} Z_i\,, \forall i = 1,2,..,5$$

$$\hat{P}\big(\tilde{Y} = max\big) \geq 0$$

$$\hat{P}\big(\ \tilde{Y} = max\big) \leq 1$$

$$\widehat{COV} \leq \ 1$$

$$\widehat{COV} \geq 0$$

$$A, B, C', D, E\ \geq 0$$

$$\widehat{Med\imath an}\ \in \{1,2, \ldots 5\}$$

By solving this model, the optimal levels of the controllable factors and COV and median values at this combination of factor levels are obtained as presented in Table 3.24.

Table 3.24 Solutions of Non-Linear Model for Surface Defect Case

A	B	C'	D	E	F	$\widehat{COV}$	$\widehat{Median}$
1	1.8	2.8	1	1	-	0.2092	1

The non-linear mathematical model gives median as 1 and COV as 0.21. Median is at the target quality level. Since COV is 0.21, the minimum dispersion level, 0, is not achieved with this result. However, it is at a reasonable level and not so high. That means, with these factor levels, a manufacturer can produce high-quality polysilicon wafers with a low variance. Also, this parameter setting quite similar to experiment 11th except factor E. So, optimal parameter design for surface defect case is $A_1B_{1.8}C'_{2.8}D_1E_1$.

Step 5: Verification of results with Random Forest model

These optimal factor levels are used in RF model to test the accuracy of the results for each class (category). Estimated class probabilities from the RF, COV and median calculated using these estimates in Equations (3.9) and (3.10) respectively are given in Table 3.25.

Table 3.25 Estimated Class Probabilities, COV and Median with Optimal Factors for Surface Defect Case

$\widehat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{I})$	$\widehat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{II})$	$\widehat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{III})$	$\widehat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{IV})$	$\widehat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{V})$	$\widehat{\boldsymbol{F_Y}}(\boldsymbol{I})$	$\widehat{\boldsymbol{F_Y}}(\boldsymbol{II})$	$\widehat{\boldsymbol{F_Y}}(\boldsymbol{III})$	$\widehat{\boldsymbol{F_Y}}(\boldsymbol{IV})$	$\widehat{\boldsymbol{F_Y}}(\boldsymbol{V})$	$\widehat{\boldsymbol{COV}}$	$\widehat{\boldsymbol{Median}}$
0.7842	0.1352	0.0493	0.0186	0.0128	0.7843	0.9194	0.9687	0.9873	1	0.1551	1

As expected, the RF model also estimates the probability of Class I as highest and the calculated COV for this probability is similar to what is obtained with the optimization. Therefore, these COV and median values are same with the mathematical model results.

The optimal factor levels and the class (category) probabilities in Table 3.25 are compared with the optimal solution of Gülbudak Dil (2018). Optimal design for Surface Defect data in Gülbudak Dil (2018) is $A_1B_{1.983}C_{2.996}D_1E_{2.989}F_2$. In this current study, these design parameters are used to estimate the class probabilities with RF. Obtained class probabilities are given in Table 3.26. The comparison of class probabilities for the proposed method and Gülbudak Dil (2018) is given in Figure 3.26.

Table 3.26 The Class Probability Estimation of Gülbudak Dil (2018) with RF for Surface Defect Data

$\widehat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{I})$	$\widehat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{II})$	$\widehat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{III})$	$\widehat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{IV})$	$\widehat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{V})$
0.8085	0.091	0.0114	0.0277	0.0615

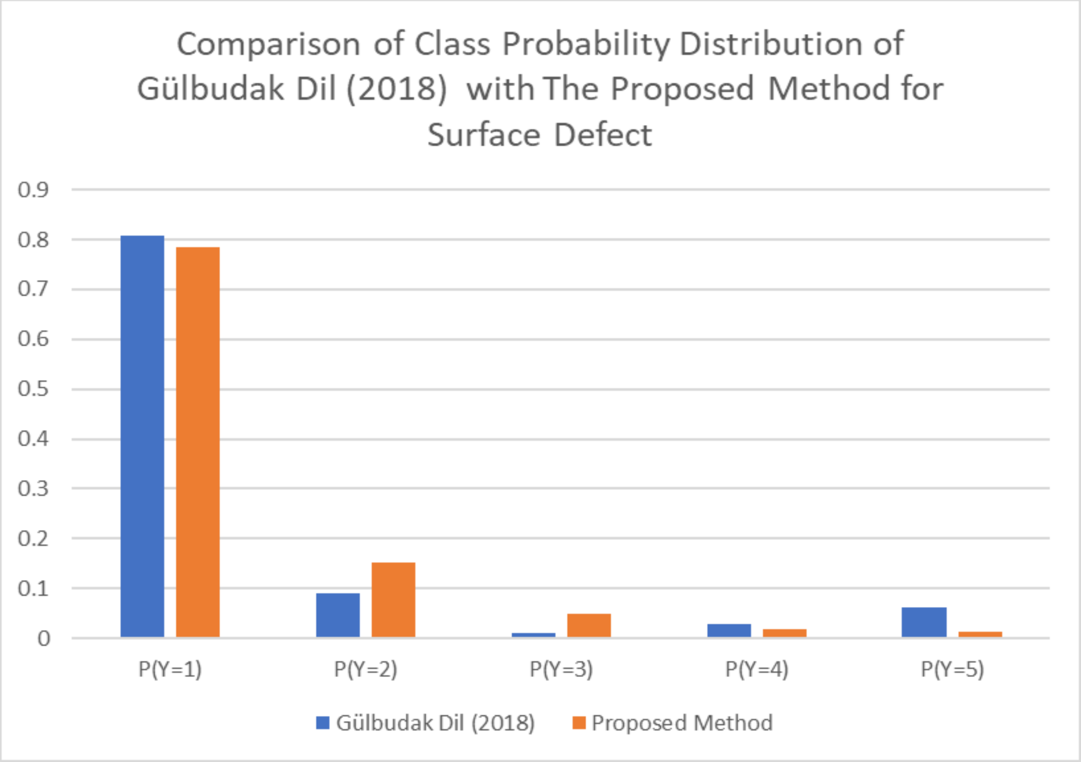


Figure 3.26 The Class Probabilities Comparison of Gülbudak Dil (2018) and The Proposed Method for Surface Defect Data

Gülbudak Dil (2018) estimates the first class with higher probability. On the other hand, probabilities mostly belong to 1st and 2nd class for the proposed method. Especially for Class V, the solution of Gülbudak Dil (2018) has higher probability than the proposed method. Since Class V is the worst category, this causes the variation. In terms of dispersion, the proposed method gives more robust solution the Gülbudak Dil (2018).

3.5 Case Study II: Inkjet Printer

This dataset is first published by Logothetis (1992). The dataset is from an experimental design to select best ink mixture for inkjet printers. The resulting ink mixture should achieve high adhesion on metal and plastic substrace. There are five substances (controllable factors): dye, carbitol, PM, resin, and water. The controllable factors and their levels are given in Table 3.27.

Step 1: Collect the experimental data

Table 3.27 Controllable Factors and Their Levels for Inkjet Printer Case (Source: (Logothetis, 1992))

Factors	**Levels**	
	0	1
1. Dye	1%	3%
2. Carbitol	1.5%	2.5%
3. PM	6.5%	9.5%
4. Resin	8%	12%
5. Water	10%	20%

The quality levels for this case are defined by rubbing. If the ink of the printed code removes and code becomes unreadable after 10 rubs, the mixture of ink is classified as low quality. On the other hand, if it becomes unreadable after 26 or more rubs it has the highest quality ink mixture, Class IV. So, this problem is larger-the-better type of problem, quality increases as categories increase. The categories (classes) and their ranges are given in Table 3.28.

Table 3.28 Classes and Their Ranges for the Inkjet Printer Case

Classes	**Range**	
	Minimum	**Maximum**
I	1	10

Table 3.28 (cont’d) Classes and Their Ranges for the Inkjet Printer Case

II	11	28
III	19	25
IV	26	∞

Logothetis (1992) uses L_8 orthogonal array, conduct eight different experiments, with ten replications. There are $8 \times 10 = 80$ observations in the dataset. For each replication, the resulting ink mixture is classified according to its range of quality. The resulting dataset is given in Table 3.29.

Table 3.29 Experimental Dataset for Inkjet Printer Case (Source: Logothetis (1992))

Exp. No	**Factors**					**Number of Observations by Classes**			
	A	B	C	D	E	I	II	III	IV
1	0	0	0	0	0	4	3	3	0
2	0	0	1	1	1	2	5	1	2
3	0	1	0	1	1	2	5	1	2
4	0	1	1	0	0	3	4	2	1
5	1	0	0	0	1	8	0	1	1
6	1	0	1	1	0	10	0	0	0
7	1	1	0	1	0	1	0	3	6
8	1	1	1	0	1	2	1	1	6

Step 2 and 3: Build Random Forest model, Calculate COV and median

Hyperparameter tuning stage is done using both OOB error and GSCV. Finding the best parameter setting is explained and resulting performance of the RF model is given in Section 3.1.2.2. The estimated class probabilities obtained with these parameters are tabulated in Table 3.30 with the resulting COV and median values.

Table 3.30 Estimated Class Probabilities, COV and Median for Inkjet Printer Case

Exp. No	$\boldsymbol{\hat{P}(Y = I)}$	$\boldsymbol{\hat{P}(Y = II)}$	$\boldsymbol{\hat{P}(Y = III)}$	$\boldsymbol{\hat{P}(Y = IV)}$	$\boldsymbol{\widehat{F_Y}(I)}$	$\boldsymbol{\widehat{F_Y}(II)}$	$\boldsymbol{\widehat{F_Y}(III)}$	$\boldsymbol{\widehat{F_Y}(IV)}$	$\boldsymbol{\widehat{COV}}$	$\boldsymbol{\widehat{Median}}$
1	0.1814	0.3485	0.4701	0	0.1813	0.5298	1	1	0.3145	2
2	0.09	0.5729	0.1572	0.1799	0.0900	0.6629	0.8201	1	0.3705	2
3	0.6252	0.2045	0	0.1703	0.6252	0.8296	0.8296	1	0.4425	1
4	0.148	0.495	0.2589	0.0981	0.1480	0.6430	0.9018	1	0.3613	2
5	0.6058	0	0.2229	0.1712	0.6058	0.6058	0.8287	1	0.5829	1
6	1	0	0	0	1	1	1	1	0	1
7	0.0506	0	0.3890	0.5606	0.0504	0.0506	0.4396	1	0.2628	4
8	0.0923	0.1202	0.1498	0.6378	0.0923	0.2125	0.3622	1	0.4023	4

According to Table 3.30, experiments 7th ($A_1B_1C_1D_0E_1$) and 8th ($A_1B_1C_0D_1E_0$) give the most preferred value for median which is Class IV but the COV values at these settings are 0.2628 and 0.4023, respectively.

Step 4: Find optimal design parameters with minimum COV and target quality level

First, the regression models for COV and median should be estimated. Similar to surface defect case, Linear regression for COV and ordinal logistic regression for median is used. According to Table 3.27, all the factors are originally continuous variables, and this is preserved while fitting regression models. For linear regression, main factors and their two-way interactions are added to the model and using the stepwise function in Minitab 20 (2020) with significance level of 0.2 to enter or remove a variable an adequate model is obtained. After that, adding significant terms and removing unsignificant ones by considering a significance level of 0.05, the final model is obtained. In the final model, R^2 and adjusted-R^2 are 99.89% and 99.62%, respectively, which are at satisfactory level. Also, all p-values are less than 0.05, which means all terms are significant. For median, both main effects and two-way interactions are added to the model but none of them are significant, and model's performance does not improve. Therefore, only Factor B is used to fit the model. P-value of the final model is 0.11 which is higher than 0.05. So, Ordinal Logistic Regression has moderate performance on this dataset. The regression performances are given and discussed in Appendix I. Resulting mathematical formulations are:

$$\widehat{COV} = 0.32337 - 0.06023 \times A + 0.03203 \times C + 0.31663 \times E - 0.2034 \times B \times E - 0.2983 \times C \times D$$

$$\hat{P}(\tilde{Y} \leq 1) = 1 - exp(-exp(-0.180644 - 1.56124 \times B))$$

$$\hat{P}(\tilde{Y} \leq 2) = 1 - exp\big(-exp(1.26228 - 1.56124 \times B)\big)$$

As it can be seen in Table 3.29, Class 3 is not observed in the prediction results. So, these cumulative probabilities only calculated for Class $1, 2$, and 4.

To find the best design among the tested and non-tested ones a non-linear optimization problem is solved using Baron Solver (2018). There are two objectives:

1. Reducing the categorical variance, minimizing the COV
2. Obtaining the highest quality level, maximizing the median

The target quality level is 4, then bias is calculated by subtracting 4 from the estimated median. The mathematical model is as follows:

$$Min\ \ 0.5 \times \left(\widehat{COV}\right)^2 + 0.5 \times \ \left(\widehat{Median} - 4\right)^2$$

s.t.

$\widehat{COV} = \ 0.32337 - 0.06023 \times A\ +\ 0.03203 \times C\ +\ 0.31663 \times\ E - 0.2034 \times B \times E - 0.2983 \times\ C \times D$

$\hat{P}(\tilde{Y} = 1) = 1 - exp(-exp(-0.180644 - 1.56124 \times B))$

$\hat{P}\left(\tilde{Y} = 2\right) = 1 - exp\left(-exp(1.26228 - 1.56124 \times B)\right) - \hat{P}\left(\tilde{Y} \le 1\right)$

$\hat{P}\left(\tilde{Y} = 3\right) = 0$

$\sum_{i=1}^{4} \hat{P}\left(\tilde{Y} = i\right) = 1$

$\hat{P}\left(\ \tilde{Y} = max\right) \le\ \hat{P}\left(\tilde{Y} = i\right) + (1 - x_i)\ \ \forall i = 1,2,3,4$

$\hat{P}\left(\ \tilde{Y} = max\right) \ge\ \hat{P}\left(\tilde{Y} = i\right)\ \ \forall i = 1, 2, 3, 4$

$\sum_{i=1}^{4} x_i = 1\ \ \forall i = 1, 2, 3, 4$

$Z_i = i \times x_i\ \forall i = 1, 2, 3, 4$

$\widehat{Median} =\ \ \sum_{i=1}^{4} Z_i$

$0 \le \hat{P}\left(\tilde{Y} = max\right) \le 1$

$0 \le\ \widehat{COV}\ \le\ 1$

$A, B, C, D, E \ge 0$

$\widehat{Median} \in \{1, 2, 3, 4\}$

By solving this model, the optimal levels of the controllable factors and COV and median values at this combination of factor levels are obtained as provided in Table 3.31.

Table 3.31 Solution of Non-Linear Model for Inkjet Printer Case

A	**B**	**C**	**D**	**E**	$\widehat{\boldsymbol{COV}}$	$\widehat{\boldsymbol{Median}}$
1	1	0.3	0.8	0.5	0.2577	4

Median is estimated as 4 and COV is 0.2577. The optimal factor levels are quite similar to experiment 7th and 8th.

Step 5: Verification of results with Random Forest model

To check consistency between what is obtained by non-linear mathematical model and RF, these factor levels are put into the RF. Class probabilities estimated by RF, resulting COV and median values calculated by Equations (3.9) and (3.10) respectively are given in Table 3.32.

Table 3.32 Estimated Class Probabilities, COV and Median with Optimal Factors for Inkjet Printer Case

$\widehat{\boldsymbol{P}}(\boldsymbol{Y}=\boldsymbol{I})$	$\widehat{\boldsymbol{P}}(\boldsymbol{Y}=\boldsymbol{II})$	$\widehat{\boldsymbol{P}}(\boldsymbol{Y}=\boldsymbol{III})$	$\widehat{\boldsymbol{P}}(\boldsymbol{Y}=\boldsymbol{IV})$	$\widehat{\boldsymbol{F_Y}}(\boldsymbol{I})$	$\widehat{\boldsymbol{F_Y}}(\boldsymbol{II})$	$\widehat{\boldsymbol{F_Y}}(\boldsymbol{III})$	$\widehat{\boldsymbol{F_Y}}(\boldsymbol{IV})$	$\widehat{\boldsymbol{COV}}$	$\widehat{\boldsymbol{Median}}$
0.0506	0	0.389	0.5604	0.0506	0.0506	0.4396	1	0.2628	4

Using these estimated probabilities from the RF model, median and COV are calculated as 4 and 0.2628, respectively. These results are parallel to the solution of the mathematical model. For some manufacturers, it can be more important to produce high quality product and therefore they have tolerance to some degree of dispersion. This result can be desirable for such inkjet producers.

The optimal factor levels and the class (category) probabilities in Table 3.32 are compared with the optimal solution of Gülbudak Dil (2018). Optimal design for Inkjet Printer data in Gülbudak Dil (2018) is $A_{0.905}B_{0.9842}C_{0.4504}D_0$. In this study, these design parameters are used to estimate the class probabilities with RF. Obtained class probabilities are given in Table 3.33. The comparison of class probabilities for the proposed method and Gülbudak Dil (2018) is given in Figure 3.27.

Table 3.33 The Class Probability Estimation of Gülbudak Dil (2018) with RF for Inkjet Printer Data

$\hat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{I})$	$\hat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{II})$	$\hat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{III})$	$\hat{\boldsymbol{P}}(\boldsymbol{Y} = \boldsymbol{IV})$
0.313	0.0671	0.1866	0.4333

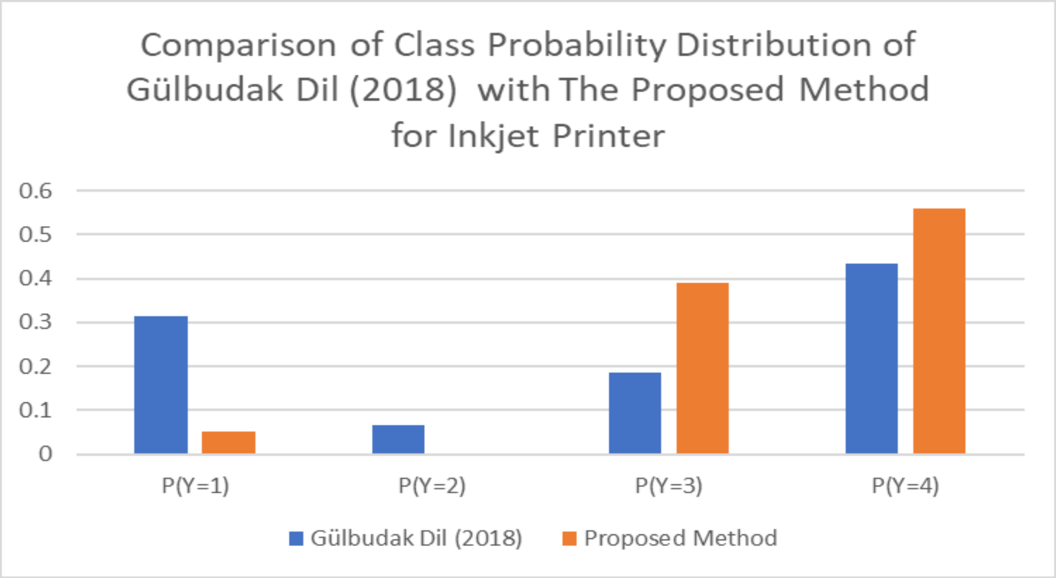


Figure 3.27 The Class Probabilities Comparison of Gülbudak Dil (2018) and The Proposed Method for Inkjet Printer Data

The proposed method estimates the target quality level (Class IV) with a higher probability than the Gülbudak Dil (2018) and the probabilities are mainly in the Class III and Class IV. However, Gülbudak Dil (2018) has higher probability for the worst class (Class I), this situation increases the variability and decreases the robustness. For this case study, the proposed method performs better in terms of probability distributions.

3.6 Case Study III: Duplicator

This dataset is collected by Logothetis and Wynn (1994) to select the best parameters that affect the paper feeding stage of the duplicator machine. It is also used by Karabulut (2013) and Gülbudak (2018) to apply different RPD methods.

Step 1: Collect the experimental data

In this experiment, paper sheets are fed to the duplicator and the aim is to select the best parameters that provide successful feeding through the duplicator. There are

12 controllable factors and one interaction of these factors is used in the data collecting step. These factors and their levels are provided in Table 3.34.

Table 3.34 Controllable Factors and Their Levels for Duplicator Case

Factors	**Levels**	
	0	**1**
A. Vacuum Header Type	Normal	Lightweight
B. Feed cam type	Normal	Smoothed
C. Master cylinder cam	Smoothed	Normal
D. Air rifle setting	Normal	High
E. Chain gripper release cam	Normal	Advanced
F. Paper Weight Bar Spring	Without	With
G. Release Blowdown Spray	OFF	ON
H. Buckle Setting	Normal	High
I. Paperweight Bar	Light	Heavy
J. Paperweight Bar Position	Normal	Back
K. Impression Roller Setting	Normal	High
L. Vacuum Setting	Normal	High

The quality level is determined by count of paper feeding and expressed as an ordered categorical variable: Class I shows low quality, that is paper feeding is failed. Class IV shows the highest quality level, it means 337 or more paper sheets are fed through a duplicator machine. So, this is a larger-the-better type of problem. The categories (classes) and their ranges are given in Table 3.35.

Table 3.35 Classes and Their Ranges for Duplicator Case

Classes	**Range**
I	Paper Feeding Failed
II	$[1, 168]$
III	$[169, 336]$
IV	$[337, \infty]$

Logothetis and Wynn (1994) uses L_{16} orthogonal design therefore 16 different parameter settings are included in the experiment. Each experiment is replicated 4 times, and the obtained quality level is recorded for these runs. There are $16 \times 4 = 64$ observations in this dataset. The resulting dataset is tabulated in Table 3.36.

Table 3.36 Experimental Dataset for Duplicator Case (Source: Logothetis and Wynn (1994))

Exp. No.	**Factors**													**Number of Observations by Classes**			
	A	B	C	D	E	F	G	H	FxI	I	J	K	L	I	II	III	IV
1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3	0	0
2	0	0	0	0	0	0	1	1	1	1	1	1	1	1	3	0	0

Table 3.36 (cont'd) Experimental Dataset for Duplicator Case (Source: Logothetis and Wynn (1994))

3	0	0	0	1	1	1	0	0	0	1	1	1	1	1	3	0	0
4	0	0	0	1	1	1	1	1	1	0	0	0	0	1	3	0	0
5	0	1	1	0	0	1	0	0	1	0	0	1	1	0	3	0	1
6	0	1	1	0	0	1	1	1	0	1	1	0	0	1	0	0	3
7	0	1	1	1	1	0	0	0	1	1	1	0	0	1	1	1	1
8	0	1	1	1	1	0	1	1	0	0	0	1	1	0	2	1	1
9	1	0	1	0	1	0	0	1	0	0	1	0	1	1	3	0	0
10	1	0	1	0	1	0	1	0	1	1	0	1	0	2	2	0	0
11	1	0	1	1	0	1	0	1	0	1	0	1	0	1	3	0	0
12	1	0	1	1	0	1	1	0	1	0	1	0	1	0	4	0	0
13	1	1	0	0	1	1	0	1	1	0	1	1	0	0	1	1	2
14	1	1	0	0	1	1	1	0	0	1	0	0	1	0	0	0	4
15	1	1	0	1	0	0	0	1	1	1	0	0	1	0	2	0	2
16	1	1	0	1	0	0	1	0	0	0	1	1	0	1	2	1	0

Step 2 and 3: Build Random Forest model, Calculate COV and median

Hyperparameter tuning stage is done using both OOB error and GSCV. Finding best parameter setting and resulting performance of the RF model is given in the beginning of this section. The estimated class probabilities obtained by RF with these parameters are given in Table 3.37 with resulting COV and median values.

Table 3.37 Estimated Class Probabilities, COV and Median for Duplicator Case

Exp. No	$\widehat{P}(Y = I)$	$\widehat{P}(Y = II)$	$\widehat{P}(Y = III)$	$\widehat{P}(Y = IV)$	$\widehat{F_Y}(I)$	$\widehat{F_Y}(II)$	$\widehat{F_Y}(III)$	$\widehat{F_Y}(IV)$	$\widehat{COV}$	$\widehat{Median}$
1	0.4015	0.5985	0	0	0.4015	1	1	1	0.1756	2
2	0.3984	0.5996	0	0.0020	0.3984	0.9980	0.9980	1	0.1784	2
3	0.3922	0.6058	0	0.0020	0.3922	0.9980	0.9980	1	0.1773	2
4	0.4114	0.5868	0.0018	0	0.4114	0.9982	1	1	0.1786	2
5	0.0036	0.6211	0.0020	0.3732	0.0036	0.6247	0.6268	1	0.3911	2
6	0.2587	0.0042	0.0063	0.7308	0.2587	0.2629	0.2692	1	0.5271	4
7	0.2365	0.1110	0.4149	0.2376	0.2365	0.3475	0.7624	1	0.5359	3
8	0.0017	0.2729	0.4625	0.2629	0.0017	0.2746	0.7371	1	0.3117	3
9	0.3923	0.6077	0	0	0.3923	1	1	1	0.1741	2
10	0.7020	0.2961	0.0018	0.0001	0.7020	0.9980	0.9999	1	0.1525	1

Table 3.37 (cont’d) Estimated Class Probabilities, COV and Median for Duplicator Case

11	0.4187	0.5813	0	0	0.4187	1	1	1	0.1781	2
12	0.0153	0.9847	0	0	0.0153	1	1	1	0.0101	2
13	0.0022	0.1182	0.4683	0.4113	0.0022	0.1204	0.5887	1	0.2699	3
14	0.0051	0.0087	0.0020	0.9843	0.0051	0.0137	0.0157	1	0.0230	4
15	0.0026	0.3502	0.0065	0.6408	0.0026	0.3527	0.3592	1	0.3793	4
16	0.2729	0.2705	0.4487	0.0079	0.2729	0.5434	0.9921	1	0.3722	2

According to Table 3.37 experiment 14th $(A_1B_1C_0D_0E_1F_1G_1H_0FxI_0I_1J_0K_0L_1)$ gives the most preferred value for median which is Class IV with a very low COV, 0.0230. After this step, empirical models for COV and median are built and a non-linear optimization problem is solved to see if there is any better design.

Step 4: Find optimal design parameters with minimum COV and target quality level

First, the empirical models of COV and median should be built as functions of control factors. According to Table 3.34 all the factors are originally categorical (nominal) variables, and this is preserved while fitting regression models. For linear regression, factors C, D, E, H, FxI, I and K are not found significant for linear regression. In the resulting model, R^2 and adjusted-R^2 are 88.68% and 83.01%, respectively. Also, all p-values less than 0.1, only the p-value of factor F is 0.069, therefore, all terms are significant, and performance of the model is at satisfactory level. For the median, the main factors are added to the model, but interactions are not found as significant. Therefore, only factor H and K is used to fit the model. All terms are significant since all p-values are less than 0.01. The regression performances are given and discussed in Appendix J. The estimated regression models are as follows.

$$\widehat{COV} = 0.2287 + 0.1982 \times B_1 - 0.0655 \times F_1 - 0.1397 \times A_1 \times G_1 + 0.1253 \times G_1 \times J_1 - 0.1397 \times A_1 \times G_1 + 0.1253 \times G_1 \times J_1 - 0.1571 \times G_1 \times L_1$$

$$\hat{P}(\tilde{Y} \leq 1) = 1 - exp(-exp(-3.35636 - 1.158215 \times H_1 + 1.59750 \times K_1))$$

$$\hat{P}(\tilde{Y} \leq 2) = 1 - exp(-exp(-0.219923 - 1.158215 \times H_1 + 1.59750 \times K_1))$$

$$\hat{P}(\tilde{Y} \leq 3) = 1 - exp(-exp(0.546434 - 1.158215 \times H_1 + 1.59750 \times K_1))$$

To find the best design among the tested and non-tested ones we solve a non-linear optimization problem by using Baron Solver (2018). There are two objectives:

1. Reducing the categorical variance, minimizing the COV
2. Obtaining the highest quality level, maximizing the median

Similar to inkjet printer case, WMSE is used to combine bias and COV in the objective function. Target quality level is 4, then bias is calculated by subtracting 4 from the estimated median. The mathematical model is as follows.

$$Min\ 0.5 \times \left(\widehat{COV} \right)^2 + 0.5 \times \left(\widehat{Median} - 4 \right)^2$$

s.t.

$\widehat{COV} = 0.2287 + 0.1982 \times B_1 - 0.0655 \times F_1 - 0.1397 \times A_1 \times G_1 + 0.1253 \times G_1 \times J_1 - 0.1397 \times A_1 \times G_1 + 0.1253 \times G_1 \times J_1 - 0.1571 \times G_1 \times L_1$

$\hat{P}(\tilde{Y} = 1) = 1 - exp(-exp(-3.35636 - 1.158215 \times H_1 + 1.59750 \times K_1))$
$\hat{P}(\tilde{Y} = 2) = 1 - exp(-exp(-0.219923 - 1.158215 \times H_1 + 1.59750 \times K_1)) - \hat{P}(\tilde{Y} \leq 1)$

$\hat{P}(\tilde{Y} = 3) = 1 - exp\left(-exp(0.546434 - 1.158215 \times H_1 + 1.59750 \times K_1)\right) - \hat{P}(\tilde{Y} \leq 2)$

$\sum_{i=1}^{4} \hat{P}(\tilde{Y} = i) = 1$

$\hat{P}(\tilde{Y} = max) \leq \hat{P}(\tilde{Y} = i) + (1 - x_i)\ \forall i = 1,2,3,4$

$\hat{P}(\tilde{Y} = max) \geq \hat{P}(\tilde{Y} = i)\ \forall i = 1,2,3,4$

$\sum_{i=1}^{4} x_i = 1\ \forall i = 1,2,3,4$

$Z_i = i \times x_i\ \forall i = 1,2,3,4$

$\widehat{Median} = \sum_{i=1}^{4} Z_i$

$0 \leq \hat{P}(\tilde{Y} = max) \leq 1$

$0 \leq \widehat{COV} \leq 1$

$A_0, A_1, B_0, B_1, F_0, F_1, G_0, G_1, H_0, H_1, J_0, J_1, K_0, K_1, L_0, L_1 \in \{0, 1\}$

$\widehat{Median} \in \{1, 2, 3, 4\}$

Solution of the model is provided in Table 3.38.

Table 3.38 Solutions of Non-Linear Model for Duplicator Case

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**FxI**	**I**	**J**	**K**	**L**	$\widehat{\boldsymbol{COV}}$	$\widehat{\boldsymbol{Median}}$
1	1	-	-	-	1	1	1	-	-	0	0	1	0.0646	4

Median is at the target level, 4, and COV is very small, 0.0646. This parameter setting is nearly same with experiment 14th except factor H. The target quality level, 4, is achieved with this result. However, COV is higher than the ideal value, 0.

Step 5: Verification of results with Random Forest model

To check consistency between what is obtained by the non-linear mathematical model and RF, class probabilities at the factor levels given in Table 3.37 are estimated again with RF. Estimated class probabilities, COV and median with these probabilities computed by Equations (3.9) and (3.10) respectively are given in Table 3.39.

Table 3.39 Estimated Class Probabilities, COV and Median with Optimal Factors for Duplicator Case

$\widehat{P}(Y = I)$	$\widehat{P}(Y = II)$	$\widehat{P}(Y = III)$	$\widehat{P}(Y = IV)$	$\widehat{F_Y}(I)$	$\widehat{F_Y}(II)$	$\widehat{F_Y}(III)$	$\widehat{F_Y}(IV)$	$\widehat{COV}$	$\widehat{Median}$
0.0127	0.0339	0.0136	0.9398	0.0127	0.0466	0.0602	1	0.0788	4

Results of non-linear optimization and the RF model are parallel to each other. Both achieve the highest quality level, and estimated COV values are really close to each other. Using these parameter designs, $A_1B_1F_1G_1H_1J_0K_0L_1$ 337 or more sheets can be fed through a duplicator, almost every time.

The optimal factor levels and the class (category) probabilities in Table 3.39 are compared with the optimal solution of Gülbudak Dil (2018). Optimal design for Duplicator data in Gülbudak Dil (2018) is $A_0B_1C_0D_1E_1F_1G_1H_1FxI_0J_1K_1L_1F_1G_1H_1J_0K_0L_1$. In this study, these design parameters are used to estimate the class probabilities with RF. Obtained class probabilities are given in Table 3.40. The comparison of class probabilities for the proposed method and Gülbudak Dil (2018) is given in Figure 3.28.

Table 3.40 The Class Probability Estimation of Gülbudak Dil (2018) with RF for Duplicator Data

$\boldsymbol{\widehat{P}(Y = I)}$	$\boldsymbol{\widehat{P}(Y = II)}$	$\boldsymbol{\widehat{P}(Y = III)}$	$\boldsymbol{\widehat{P}(Y = IV)}$
0.1048	0.1421	0.2942	0.4229

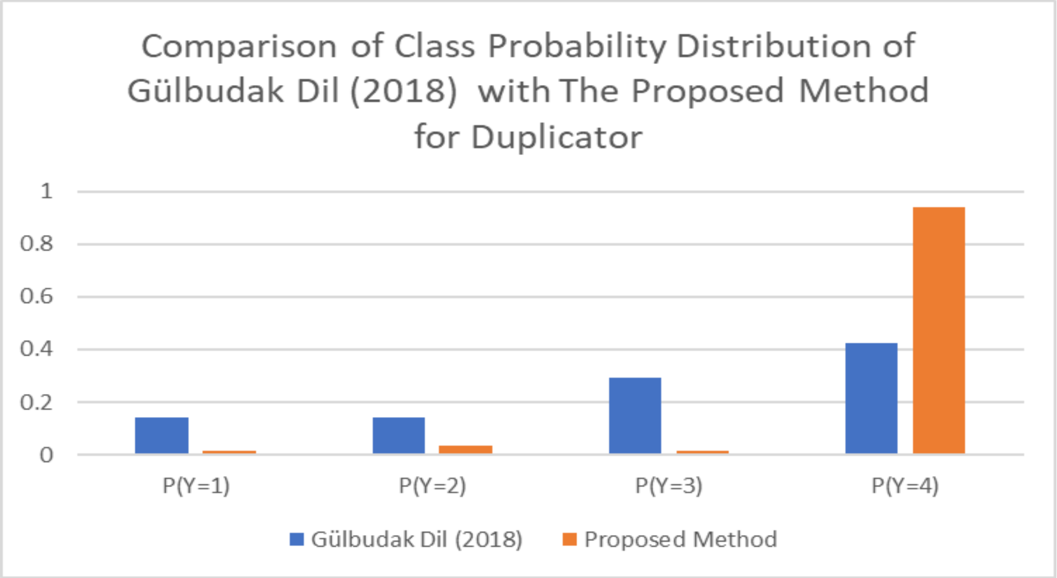


Figure 3.28 The Class Probabilities Comparison of Gülbudak Dil (2018) and The Proposed Method for Duplicator Data

The proposed method estimates target quality level (Class IV) with a higher probability and the other class probabilities are much lower than the Gülbudak Dil (2018). The estimated probability for the worst class is much higher than the proposed method. In this case, the optimal design parameters obtained with the proposed method causes less variation than the Gülbudak Dil (2018).

CHAPTER 4

DISCUSSION

In Chapter 3, the best data mining algorithm is tried to be selected. To do this, hyperparameter tuning for each model is done and then repeated stratified 3- fold cross-validation is used, and test and train results are compared. Moreover, TOPSIS and Multi-MOORA methods are applied to the average of test results to rank XGBoost, Random Forest, and Logistic Regression. Four datasets are included to this study. However, applying these methods to more experimental datasets which have different features such as: number of factors, number of experiments, number of repetitions, class imbalance ratio, and number of classes will provide more accurate understanding about the algorithms.

In this study, the proposed method is applied to three different case studies which have an ordinal categorical response. Median and COV are used as location and dispersion measures, respectively. First, a Random Forest model is fitted with appropriate hyperparameter setting and then estimated class probabilities are obtained. COV and median are calculated using them. Then, regression models to estimate median and COV at any settings of controllable factors are built and a non-linear optimization problem is solved to find the controllable factor levels considering both the location (measured by median) and dispersion (measured by COV) of the response.

The proposed method is useful when the response has a categorical order. For some of the RDP studies, response can be nominal and the type of problem is nominal-the-best. Using median for such problems is not meaningful, instead using mode as a location measure will provide better results. Furthermore, for this type of problem using coefficient of nominal variation (CNV) as a dispersion measure will be more appropriate than the COV (Kvälseth, 1995a). CNV is calculated as follows.

$$CNV = 1 - \left(\frac{1}{n-1} \sum_{1 \le i} \sum_{< j \le n} \left| p_i - p_j \right|^2 \right)^{\frac{1}{2}} \quad (4.1)$$

In RPD studies, there are two common objectives: minimizing the variance and minimizing the bias between response and target value. In the proposed method, WMSE is used to combine these two objectives into a single objective function.

$$w_{Median} \times (Response - Target)^2 + w_{COV} \times (COV)^2 \quad (4.2)$$

$$w_{Median} + w_{COV} = 1 \quad (4.3)$$

Since achieving the target value and minimizing the COV is equally important in this study, equal weights are given to both objectives. However, for some users it can be more important to achieve the lowest variability and they may prefer to produce the lowest quality level or vice versa. For such cases, users can define the importance levels of COV and median, and the weights in the objective function can be arranged according to their preference. Also, Ding et al. (2007) proposed a data-driven approach for finding weights in WMSE objective function for dual response surface optimization. First, marginal optimizations are conducted for each objective and ideal points are found. Then, weight of bias is increased slowly from 0 to 1 and each obtained solution is recorded. An efficiency curve is plotted with these solutions and the weight of the solution close to the ideal solution is selected as the weight.

To solve non-linear optimization problem, the empirical models of COV and median are needed. The least squares regression and Ordinal Logistic Regression are used to find them. However, only point estimate is considered in this study. Typically, in many RPD studies, uncertainty in noise factors and response models are not considered. There might be estimation error since experimental datasets can be insufficient for some cases. So, problem about using the point estimate in RPD study is that optimal solution obtained from the estimated model may not reflect the true model (Ouyang et al., 2016). There are some studies considering interval instead of point estimates (He et al., 2021; He et al., 2012; Ouyang et al., 2016; Xu & Albin, 2003). He et al., (2012), define a robust desirability function and they try to obtain robustness by using the confidence region. They mentioned that one strength of this approach is, instead of a point estimate, they considered all values in the confidence intervals. These confidence regions are constructed based on standard error of the estimated response. After the confidence regions are defined, robust desirability function is constructed. He et al., (2021), conduct a similar study. They keep the robust desirability approach as in the He et al., (2012), solve the optimization problem interactively. It is an iterative method, in each iteration current solution is evaluated by DM.

Using Ordinal Logistic Regression to estimate the regression model of median has some limitations. The drawbacks of Logistic Regression are discussed in Section 2. For the future studies, instead of Ordinal Logistic Regression, Multivariate Adaptive Regression Splines (MARS) can be useful in this stage. MARS is a nonparametric method. So, it is not based on specific assumptions. It is a combination of linear regression, the mathematical construction of splines, binary recursive partitioning, and intelligent algorithms. The main advantages of MARS are listed as; generating simple and easily interpreted models, performing analysis on parameter relative importance, and capturing the patterns in high-dimensional data (Zhang & Goh, 2016). In a RPD study, using a non-parametric and easy-to-interpret method may provide better results than Logistic Regression.

CHAPTER 5

CONCLUSION AND FURTHER STUDIES

In this study, a new method is proposed for the RPD problem with an ordinal categorical response. According to related literature mentioned in Section 2 and the analysis in Appendix A, median and COV are selected as location and dispersion measures, respectively.

The performance of Random Forest, XGBoost, and Logistic Regression are compared in the Section 3. For all methods, related hyperparameters are tuned and resulting performance of the methods are evaluated with several classification metrics using repeated stratified 3-fold cross validation. According to these results Random Forest performs better than the others. Then, TOPSIS and Multi-MOORA methods are applied to rank these algorithms and select the best one among them. Since these datasets are imbalanced, precision, recall and AUC, which are the metrics that are highly recommended for imbalance data situations, are used. As a weighting method for both MCDM methods, two different strategies are employed: equal weighting and entropy weighting. Results for both TOPSIS and Multi-MOORA are parallel to each other. A sensitivity analysis is conducted with the TOPSIS results. According to these, Random Forest and XGBoost are better than Logistic Regression. Both can be used in RPD study. However, Random Forest is selected because it has fewer hyperparameters to optimize and it is easy to interpret.

In the proposed method, with Random Forest, estimated class probabilities are obtained. Using these probabilities, estimated COV and median are calculated for each experimental trial.

The empirical models for COV and median in terms of factors are constructed using linear regression and Ordinal Logistic Regression, respectively. Then, a non-linear optimization problem, which is also tailored to the categorical data, is solved to find optimal factor levels which are close to target response with minimum COV. As a verification method of the optimal solution, optimal factor levels are put into the RF model and class probabilities are estimated this way. Then, COV and median are calculated again with these probabilities and compared with the optimization results.

The performance of the proposed method is shown in three different case studies. One of them is a smaller-the-better type problem and other two are larger-the-better type problems. For all cases, the method found the target response value. For surface defect case, optimal response is found as 4 with a COV value of 0.2092, which means target quality level achieved with small level dispersion. In duplicator case, resulting COV is close to 0, that means the target is achieved with minimum dispersion. On the other hand, for the inkjet case, the obtained COV with optimal factor level is greater than the other two studies. In overall, the proposed method shows good performance on these three case studies.

Using confidence intervals instead of point estimate for the sake of uncertainty of the estimated models and using MARS algorithm instead of Ordinal Logistic Regression can be further research directions (see Section 4). Also, in dual response optimization studies, there are posterior (D. Lee, Jeong, et al., 2016) methods that apply algorithms first and the optimal results are found using DM’s preference at the end of the algorithm. Similarly, there are interactive approaches (D. Lee, Kim, et al., 2016; D. H. Lee & Kim, 2012), which use DM’s preference to iterate algorithms and find optimal factor levels. These approaches can be used in non-linear optimization steps for the future studies.

Interest in robust parameter design is increasing day by day. However, there are few studies considering categorical responses. The expected value and variance which are appropriate for the interval or ratio scaled data are used to analyze ordinal categorical for the RPD studies in the past. The contribution of this M.Sc. thesis is using measures which are appropriate to ordinal data and applying the proposed method to different case studies. Also, performance of different data mining algorithms is evaluated with four experimental datasets, and Random Forest is found to be easy to apply and generating meaningful results for RPD studies. The proposed method can be applied to any dataset with an ordinal categorical response.

REFERENCES

Allaj, E. (2018). Two simple measures of variability for categorical data. *Journal of Applied Statistics, 45*(8), 1497–1516. doi.org/10.1080/02664763.2017.1380787

Anaklı, Z. (2009). *A comparison of data mining methods for prediction and classification types of quality problems*/ Unpublished master's thesis, Middle East Technical University, Ankara, Turkey.

Asiabar, M. H., & Ghomi, S. M. T. F. (2007). Analysis of Ordered Categorical Data Using Expected Loss Minimization. *Quality Engineering, 18*(2), 117–121. doi.org/10.1080/08982110600567467

Balali, V., Zahraie, B., & Roozbahani, A. (2014). A comparison of AHP and PROMETHEE family decision making methods for selection of building structural system. *American Journal of Civil Engineering and Architecture, 2*(5), 149–159. doi.org/10.12691/ajcea-2-5-1

Bentéjac, C., Csörgő, A., & Martínez-Muñoz, G. (2021). A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review 54*(3) doi.org/10.1007/s10462-020-09896-5

Berry, K. J., & Mielke, P. W. (1992a). Assessment of variation in ordinal data. *Perceptual and Motor Skills, 74*(1), 63–66. doi.org/10.2466/PMS.1992.74.1.63

Berry, K. J., & Mielke, P. W. (1992b). Indices of ordinal variation. *Perceptual and Motor Skills, 74*(2), 576–578. doi.org/10.2466/PMS.1992.74.2.576

Box, G., & Jones, S. (1986). Testing in industrial experiments with ordered categorical data: discussion. *Technometrics, 28*(4), 301. doi.org/10.2307/1268976

Blair, J., & Lacy, M. G. (1996). Measures of variation for ordinal data as functions of the cumulative distribution. *Perceptual and Motor Skills, 84*(2), 411–418. doi.org/10.2466/PMS.1996.82.2.411

Brans, J. P., & Vincke, PH. (1984). A Preference Ranking Organisation Method: (The PROMETHEE Method for Multiple Criteria Decision-Making). *Management and Science, 31*(6)

Brauers, W. K. M., & Zavadskas, E. K. (2012). Robustness of MULTIMOORA: A method for multi-objective optimization. *Informatica, 23*(1), 1–25. doi.org/10.15388/informatica.2012.346

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*, 123–140. doi.org/10.3390/risks8030083

Breiman, L. (2001). Random Forests. *Machine Learning, 45*(1), 5–32. doi.org/10.1023/A:1010933404324

Brodersen, K. H., Ong, C. S., Stephan, K. E., & Buhmann, J. M. (2010). The balanced accuracy and its posterior distribution. *20th International Conference on Pattern Recognition*, 3125–3128. doi.org/10.1109/ICPR.2010.764

Brownlee, J. (2020). *Imbalanced Classification with Python - Choose Better Metrics, Balance Skewed Classes, and Apply Cost-Sensitive Learning*, Machine Learning Mastery.

Casella, G., & Berger, R. (2002). *Statistical Inference, 2nd Ed.,* Duxbury Thomson Learning. Belmont.

Ceballos, B., Lamata, M. T., & Pelta, D. A. (2016). A comparative analysis of multi-criteria decision-making methods. *Progress in Artificial Intelligence*, *5*(4), 315–322. doi.org/10.1007/s13748-016-0093-1

Chakraborty, S. (2011). Applications of the MOORA method for decision making in manufacturing environment. *International Journal of Advanced Manufacturing Technology, 54*(9), 1155–1166. doi.org/10.1007/s00170-010-2972-0

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*, 785–794. doi.org/10.1145/2939672

Chipman, H., & Hamada, M. (1996). Bayesian analysis of ordered categorical data from industrial experiments. *Technometrics, 35*(1), 10. doi.org/10.2307/1268898

Çelik, Ö., & Osmanoğlu, U. (2019). Comparing to techniques used in customer churn analysis. *Journal of Multidisciplinary Developments*, *4*(1), 30–38.

Ding, R., Lin, D. K. J., & Wei, D. (2007). Dual-Response Surface Optimization: A Weighted MSE Approach. *Quality Engineering,16*(3), 377–385. doi.org/10.1081/QEN-120027940

Dwidarma, R., Permai, S. D., & Harefa, J. (2021). Comparison of Logistic Regression and XGBoost for predicting potential debtors. *2nd International Conference on Artificial Intelligence and Data Sciences, AiDAS 2021*. doi.org/10.1109/AIDAS53897.2021.9574350

Erdural, S. (2006). *A method for Robust Design of Products or Processes with Categorical Response/* Unpublished master's thesis, Middle East Technical University, Ankara, Turkey.

Erişkin, L., Dolgun, L. E., & Köksal, G. (2021). A method for robust design of products or processes with categorical response. *Quality Engineering, 33*(3), 474-486. doi.org/10.1080/08982112.2021.1896732

Fernández, A., García, S., Herrera, F., & Chawla, N. V. (2018). SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary. *Journal of Artificial Intelligence Research, 61,* 863-905. doi.org/10.1613/jair.1.11192

Freund, Y., & Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences, 55*(1), 119–139. doi.org/10.1006/jcss.1997.1504

Freund, Y., & Schapire, R. E. (1999). A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence, 14*(5), 771–780.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, *29*(5), 1189–1232. doi.org/10.1214/aos/1013203451

Geng, M. (2006). *A Comparison of Logistic Regression to Random Forests for Exploring Differences in Risk Factors Associated with Stage at Diagnosis Between Black and White Colon Cancer Patients/* Unpublished master’s thesis. University of Pittsburgh, PA, USA.

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn and TensorFlow, 2nd Ed.,* O’Reilly Media, Inc, USA.

Gülbudak Dil, S. (2018). *Robust Parameter Design of Products and Processes with An Ordinal Categorical Response Using Random Forests /* Unpublished master’s thesis, Middle East Technical University, Ankara, Turkey.

Hafezalkotob, A., Hafezalkotob, A., Liao, H., & Herrera, F. (2019). An overview of MULTIMOORA for multi-criteria decision-making: Theory, developments, applications, and challenges. *Information Fusion, 51* 145–177. doi.org/10.1016/j.inffus.2018.12.002

Haithm, A., Saleh, A. Y., & Odabaş, A. (2021). Comparison of Gradient Boosting Decision Tree Algorithms for CPU Performance. *Journal of Institute of Science and Technology, 37*(1), 157–168.

Hand, D. J. (2007). Principles of data mining. *Drug Safety 30*(7). doi.org/10.2165/00002018-200730070-00010

He, H., & Ma, Y. (2013). *IMBALANCED LEARNING: Foundations, Algorithms, and Applications, 1st Ed.,* IEEE Press, Wiley.

He, Y., He, Z., Kim, K. J., Jeong, I. J., & Lee, D. H. (2021). A Robust Interactive Desirability Function Approach for Multiple Response Optimization Considering Model Uncertainty. *IEEE Transactions on Reliability, 70*(1), 175–187. doi.org/10.1109/TR.2020.2995752

He, Z., Zhu, P. F., & Park, S. H. (2012). A robust desirability function method for multi-response surface optimization considering model uncertainty.*European Journal of Operational Research, 221*(1), 241–247. doi.org/10.1016/J.EJOR.2012.03.009

Hosmer, D. W., & Lemeshow, S. (2000). *Applied Logistic Regression, 2nd Ed.,* Wiley.

Hossin, M., & Sulaiman, M. (2015). A Review on Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process, 5*(2), 01–11. doi.org/10.5121/ijdkp.2015.5201

Ishizaka, A., & Nemery, P. (2013). *Multi-Criteria Decision Analysis: Methods and Software,* Wiley.

Jeng, Y.-C., & Guo, S.-M. (1996). Quality Improvement for Rc06 chip resistor. *Quality And Reliability Engineering International, 12*, 439–445. doi.org/10.1002/(SICI)1099-1638(199611)12:6

Jhaveri, S., Khedkar, I., Kantharia, Y., & Jaswal, S. (2019). Success prediction using Random Forest, Catboost, XGBoost and Adaboost for kickstarter campaigns. *Proceedings of the 3rd International Conference on Computing Methodologies and Communication, ICCMC 2019,* 1170–1173. doi.org/10.1109/ICCMC.2019.8819828

Jinks, J. (1987). Reduction of Voids in a Urethane-Foam Product. *Fifth Symposium on Taguchi Methods*, 135–148.

Jiří, M. (2018). The robustness of TOPSIS results using sensitivity analysis based on weight tuning. *IFMBE Proceedings, 68*(2), 83–86. doi.org/10.1007/978-981-10-9038-7_15

Kabiraj, S., Raihan, M., Alvi, N., Afrin, M., Akter, L., Sohagi, S. A., & Podder, E. (2020). Breast Cancer Risk Prediction using XGBoost and Random Forest Algorithm. *2020 11th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2020*. doi.org/10.1109/ICCCNT49239.2020.9225451

Kackar, R. N. (1985). Off-Line Quality Control, Parameter Design, and the Taguchi Method. *Journal of Quality Technology, 17*(4), 176–188. doi.org/10.1080/00224065.1985.11978964

Karabulut, B.G. (2013). *Comparison of methods for robust parameter design of products and processes with an ordered categorical response /* Unpublished master's thesis, Middle East Technical University, Ankara, Turkey.

Kidando, E., Kitali, A. E., Kutela, B., Ghorbanzadeh, M., Karaer, A., Koloushani, M., Moses, R., Ozguven, E. E., & Sando, T. (2021). Prediction of vehicle occupants injury at signalized intersections using real-time traffic and signal data. *Accident Analysis and Prevention, 149*. doi.org/10.1016/j.aap.2020.105869

Köksal, G., Erdural, S., & İlk, Ö. (2006). A Method for Analysis of Categorical Data for Robust Product and Process Design. *Proceedings in Computational Statistics, 17th Symposium of the IASC,* Rome, Italy, Rizzi, A. and Vichi, M. (Eds.) Physica-Verlag, 573- 580.

Kumar, A., & Jain, M. (2020). Ensemble Learning for AI Developers. In C. S. John, L. Berendson, & A. Mirashi (Eds.), *Ensemble Learning for AI Developers, 1st Ed.,* Apress.

Kvålseth, T. O. (1995a). Comment on the coefficient of ordinal variation. *Perceptual and Motor Skills, 81*(2), 621–622. doi.org/10.1177/003151259508100251

Kvålseth, T. O. (1995b). Coefficients of variation for nominal and ordinal categorical data. *Perceptual and Motor Skills, 80*(3), 843–847. doi.org/10.2466/PMS.1995.80.3.843

Kvålseth, T. O. (2011). Variation for categorical variables. *International Encyclopedia of Statistical Science*. Springer, Berlin, Heidelberg.

Laerd Statistics (2020). Pearson's product moment correlation. *Statistical tutorials and software guides*. Retrieved July 15, 2022 from https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php

Landis, J. R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data for categorical of observer agreement the measurement. *Biometrics, 33*(1), 159–174.

Leik, R. K. (1966). A Measure of Ordinal Consensus. *Sociological Perspectives, 9*(2), 85–90. doi.org/10.2307/1388242

Lee, D. H., & Kim, K. J. (2012). Interactive weighting of bias and variance in dual response surface optimization. *Expert Systems with Applications, 39*(5), 5900–5906. doi.org/10.1016/j.eswa.2011.11.114

Lee, D., Jeong, I., Kim, K., Lee, D., Jeong, I., & Kim, K. (2016). A posterior preference articulation approach to dual-response-surface optimization. *IIE Transactions, 8830*. https://doi.org/10.1080/07408170903228959

Lee, D., Kim, K., & Köksalan, M. (2016). An interactive method to multiresponse surface optimization based on pairwise comparisons. *IIE Transactions, 8830*, 12–26. doi.org/10.1080/0740817X.2011.564604

Li, X., Wang, K., Liuz, L., Xin, J., Yang, H., & Gao, C. (2011). Application of the entropy weight and TOPSIS method in safety evaluation of coal mines. *Procedia Engineering, 26,* 2085–2091. doi.org/10.1016/j.proeng.2011.11.2410

Logothetis, N. (1992). *Managing For Total Quality*. Prentice-Hall.

MathWorks. (2018). MATLAB (2018). https://www.mathworks.com/

McCormick, K. (Consultant), Salcedo, J., Peck, J., & Wheeler, A. (2017). *SPSS statistics for data analysis and visualization.* John Wiley & Sons.

Minitab 20 Statistical Software. (2020). [Computer Software]. State Collage, PA:

Minitab Inc. Retrieved from www. minitab.com

Meng, H., Wang, X., & Wang, X. (2018). Expressway Crash Prediction based on Traffic Big Data. *ACM International Conference Proceeding Series, 11*–16. doi.org/10.1145/3297067.3297093

Montgomery, D. C. (2020). *Introduction to Statistical Quality Control*. Wiley, New York, USA.

Mullick, S. S., Datta, S., Dhekane, S. G., & Das, S. (2020). Appropriateness of performance indices for imbalanced data classification: An analysis. *Pattern Recognition, 102*. doi.org/10.1016/j.patcog.2020.107197

Nair, V. N. (1986). Testing in Industrial Experiments with Ordered Categorical Data. *Technometrics, 28*(4), 283. doi.org/10.2307/1268974

Opricovic, S., & Tzeng, G. H. (2004). Compromise solution by MCDM methods: A comparative analysis of VIKOR and TOPSIS. *European Journal of Operational Research, 156*(2), 445–455. doi.org/10.1016/S0377-2217(03)00020-1

Orme, J. G., & Combs-Orme, T. (2009). *Multiple Regression with Discrete Dependent Variables. In Multiple Regression with Discrete Dependent Variables*. Oxford University Press.

Ouyang, L., Ma, Y., Byun, J. H., Wang, J., & Tu, Y. (2016). An interval approach to robust design with parameter uncertainty. *International Journal of Production Research, 54*(11), 3201–3215. doi.org/10.1080/00207543.2015.1078920

Pedregosa, F., Varoquaux, G., Gramfort, A., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passsos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research, 12,* 2825–2830.

Phadke, M. S. (1989). *Quality Engineering Using Robust Design*. Prentice- Hall.

Pohekar, S. D., & Ramachandran, M. (2004). Application of multi-criteria decision making to sustainable energy planning - A review. *Renewable and Sustainable Energy Reviews, 8*(4), 365–381. doi.org/10.1016/j.rser.2003.12.007

Python. (2021). Python: A dynamic, open-source programming language. Retrieved from https://www.python.org/

Quevedo, R. P., Maciel, D. A., Uehara, T. D. T., Vojtek, M., Renno, C. D., Pradhan, B., Vojtekova, J., & Pham, Q. B. (2021). Consideration of spatial heterogeneity in landslide susceptibility mapping using geographical random forest model. *Geocarto International*. doi.org/10.1080/10106049.2021.1996637

Rabby, Y. W., Hossain, M. B., & Abedin, J. (2020). Landslide susceptibility mapping in three Upazilas of Rangamati hill district Bangladesh: application and comparison of GIS-based machine learning methods. *Geocarto International,* 1–27. doi.org/10.1080/10106049.2020.1864026

Robinson, T. J., Borror, C. M., & Myers, R. H. (2004). Robust parameter design: A review. *Quality and Reliability Engineering International, 20*(1), 81–101. doi.org/10.1002/QRE.602

Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review, 33(*2), 1–39. doi.org/10.1007/s10462-009-9124-7

Rokach, L. (2019). *Ensemble Learning: Pattern Classification Using Ensemble Methods, Series in Machine Perception and Artificial Intelligence, 2nd Ed.,* World Scientific Publishing.

Roy, B. (1996). Multicriteria Methodology for Decision Aiding. *Nonconvex Optimization and Its Applications, 12*. doi.org/10.1007/978-1-4757-2500-1

Sagi, O., & Rokach, L. (2018). Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8*(4). doi.org/10.1002/WIDM.1249

Senthan, P., Rathnayaka, R. M. K. T., Kuhaneswaran, B., & Kumara, B. T. G. S. (2021). Development of churn prediction model using XGBoost - Telecommunication Industry in Sri Lanka. *2021 IEEE International IOT, Electronics and Mechatronics Conference, IEMTRONICS 2021 - Proceedings*. doi.org/10.1109/IEMTRONICS52119.2021.9422657

Stevens, S. S. (1946). On the theory of scales of measurement. *Science, 103*(2684), 677–680.

Taguchi, G. (1974). A new statistical analysis for clinical data, the accumulating analysis, in contrast with the Chi-Square test. *Suishin Igaku, 29*, 806–813.

Tang, F., & Ishwaran, H. (2017). Random forest missing data algorithms. *Statistical Analysis and Data Mining, 10(*6), 363–377. doi.org/10.1002/sam.11348

Tattar, P. N. (2018). *Hands-on Ensemble Learning with R*. Packt Publishing.

Weisberg, H.F. (1992). Central tendency and variability. Sage University Paper Series No. 07-083. Sage Publications, Newbury Park, CA.

Weiß, C. H. (2019a). On some measures of ordinal variation. *Journal of Applied Statistics, 46*(16), 2905–2926. doi.org/10.1080/02664763.2019.1620707

Weiß, C. H. (2019b). Distance-Based Analysis of Ordinal Data and Ordinal Time Series. *Journal of the American Statistical Association, 115*(531), 1189–1200. doi.org/10.1080/01621459.2019.1604370

Wu, F. C., & Yeh, C. H. (2006). A comparative study on optimization methods for experiments with ordered categorical data. *Computers & Industrial Engineering, 50*(3), 220–232. doi.org/10.1016/J.CIE.2006.04.001

XGBoost Developers. (2020). XGBoost Parameters — XGBoost 1.5.0-dev documentation. https://xgboost.readthedocs.io/en/latest/parameter.html

Xie, X., Yang, M., Xie, S., Wu, X., Jiang, Y., Liu, Z., Zhao, H., Chen, Y., Zhang, Y., & Wang, J. (2021). Early Prediction of Left Ventricular Reverse Remodeling in First-Diagnosed Idiopathic Dilated Cardiomyopathy: A Comparison of Linear Model, Random Forest, and Extreme Gradient Boosting. *Frontiers in Cardiovascular Medicine, 8*, 1–14. doi.org/10.3389/fcvm.2021.684004

Xu, D., & Albin, S. L. (2003). Robust optimization of experimentally derived objective functions. *IIE Transactions (Institute of Industrial Engineers), 35*(9), 793–802. doi.org/10.1080/07408170304408

Yoo, W., Ference, B. A., Cote, M. L., & Schwartz, A. (2012). A Comparison of Logistic Regression, Logic Regression, Classification Tree, and Random Forests to Identify Effective Gene-Gene and Gene-Environmental Interactions. *International Journal of Applied Science and Technology, 2(*7), 268.

Zhang, W., & Goh, A. T. C. (2016). Multivariate adaptive regression splines and neural network models for prediction of pile drivability. *Geoscience Frontiers, 7(*1)*,* 45–52. doi.org/10.1016/j.gsf.2014.10.003

Zhou, Z.-H. (2012). *Ensemble Methods Foundations and Algorithms, 1st Ed.,* Chapman & Hall/CRC Machine Learning & Pattern Recognition Series Ensemble Chapman and Hall/CRC.

APPENDICES

A. THE ANALYSIS OF LOCATION AND DISPERSION MEASURES

A study is carried out to observe the behavior of dispersion measures (namely LOV, Δ^*, $COV, LSQ, OV_1, OV_2, OV_{2.5}, OV_3, V(I), disp_{O,2}$) and location measures (rounded-$E[I]$, median and expected value) in different probability distributions. The aim is to observe whether dispersion and location measures can separate smaller-the-better or larger-the-better type of data. As mentioned in the literature review, LOV and COV are measures which belong to the $\widehat{OV}_\alpha$ family, where $\alpha = 1$ and $\alpha = 2$, respectively. Weiß (2019a) examined different values of α for $\widehat{OV}_\alpha$ and suggested taking α between $[2, 3]$. Therefore, $OV_{2.5}$ and OV_3 are added to the study. The variance is also included in this study for comparison. In order to see the effects of different category numbers on location and dispersion, the study is repeated where the number of categories, m, is from 2 to 10. Dispersion and location measures are examined under three different cases. One of the cases is the one that probabilities are in two extreme categories $p_1 = p$, $p_m = 1 - p$. Other is the one that probabilities are seen equally in all categories except one category as $p_i = 0\ i \epsilon \{1, \dots, m\}$, $p_j = \frac{1}{m-1} \forall j \in \{1, \dots m\} \setminus i$. Final case is the one that probabilities are seen equally in the two categories, as $p_i = p_j = 0.5\ \forall j \in\ \{1, \dots m\} \setminus i$. To illustrate the results clearly, all these cases are explained where the number of categories is $m = 4$.

The general results obtained regarding the measures are as follows. The minimum dispersion value is obtained in the one-point distribution case, where probability mass function (PMF) is $p_i = 1$, and $p_j = 0, \forall j \in\ \{1, \dots m\} \setminus i$. Minimum dispersion is 0 for LOV, Δ^*, $COV, LSQ, OV_1, OV_2, OV_{2.5}, OV_3, V(I), disp_{O,2}$ and variance, 1 for LSQ. In the case of extreme two-point distribution, where probability mass function is $p_1 = p_m = 0.5$, the maximum dispersion is obtained. Maximum dispersion is 1 for LOV, Δ^*, $COV, LSQ, OV_1, OV_2, OV_{2.5}, OV_3$ and 0 for LSQ. As the number of categories increases, dispersion values decrease for the probability mass function $p_i = 0.5$ and $p_j = 0.5\ \forall j \in \{1, \dots m\} \setminus i$ except for the extreme two-point distribution. It is observed that $V(I)$ produces the same results as variance and they are equal to $disp_{O,2} = 2V(I)$. Also, if $E[I]$ is not rounded, it provides the same results as expected value.

The cases in which the probability is observed in two extreme categories are examined. Probability distribution by categories for the case where the number of categories is $m = 4$ is shown in Table A.1.

Table A. 1 PMF of Two-point Distribution Cases, where $m = 4$

	Category-1	**Category-2**	**Category-3**	**Category-4**
Case-1	1	0	0	0
Case-2	0.9	0	0	0.1
Case-3	0.8	0	0	0.2
Case-4	0.7	0	0	0.3
Case-5	0.6	0	0	0.4
Case-6	0.5	0	0	0.5
Case-7	0.4	0	0	0.6
Case-8	0.3	0	0	0.7
Case-9	0.2	0	0	0.8
Case-10	0.1	0	0	0.9
Case-11	0	0	0	1

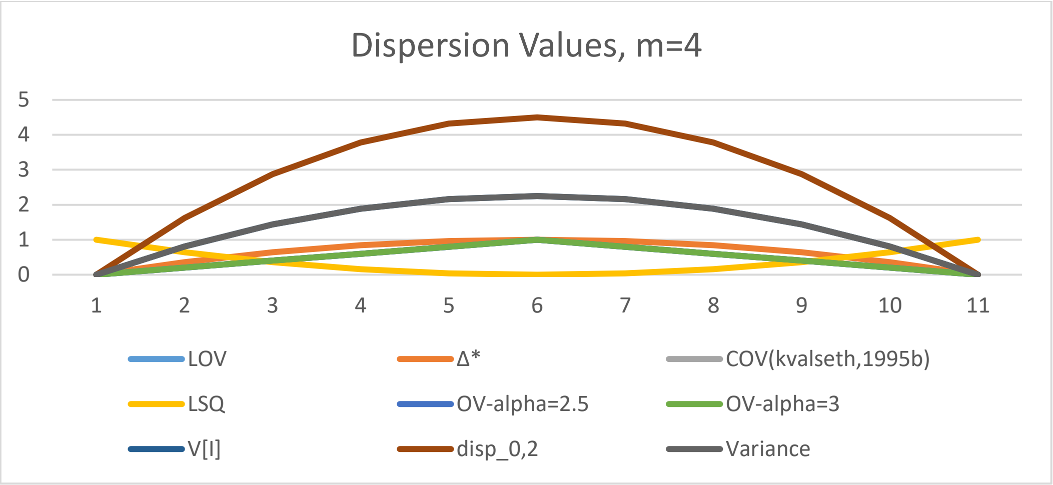


Figure A. 1 Dispersion Values Obtained with the Probability Distributions in Table A.1

Table A.2 Dispersion Values Obtained with the Probability Distributions in Table A.1

Cases	**LOV**	Δ^*	**COV**	**LSQ**	$\boldsymbol{OV_{2.5}}$	$\boldsymbol{OV_3}$	**V[I]**	$\boldsymbol{disp_{0,2}}$
Case-1	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Case-2	0.20	0.36	0.20	0.64	0.20	0.20	0.36	1.62
Case-3	0.40	0.64	0.40	0.36	0.40	0.40	0.64	2.88
Case-4	0.60	0.84	0.60	0.16	0.60	0.60	0.84	3.78
Case-5	0.80	0.96	0.80	0.04	0.80	0.80	0.96	4.32
Case-6	1.00	1.00	1.00	0.00	1.00	1.00	1.00	4.50
Case-7	0.80	0.96	0.80	0.04	0.80	0.80	0.96	4.32
Case-8	0.60	0.84	0.60	0.16	0.60	0.60	0.84	3.78
Case-9	0.40	0.64	0.40	0.36	0.40	0.40	0.64	2.88
Case-10	0.20	0.36	0.20	0.64	0.20	0.20	0.36	1.62
Case-11	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00

The probabilities in Table A.1 are symmetrical with each other. It can be seen in the Figure A.1 and Table A.2 that dispersion measures give the same result for cases where probabilities are symmetrical, e.g., dispersion of Case-2 is equal to Case-10. The measures of the $\widehat{OV}_\alpha$ family, $LOV,\ COV,\ OV_{2.5}, OV_3$ provides equal dispersion for all cases. The location values calculated for the cases given in Table A.1 are given in Figure A.2.

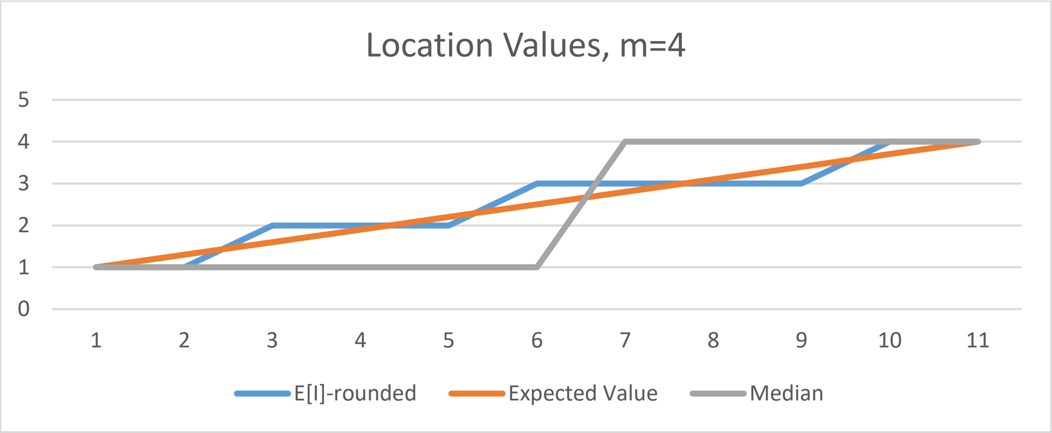


Figure A.2 Location Values Obtained with The Probability Distributions in Table A.1

The expected value shows a linear increase as seen in Figure A.1. The rounded-$E[I]$ increases more frequently, while the median increases with longer intervals. The probabilities that are seen equally in all categories except one category are discussed, as well. When the number of categories is $m = 4$, probabilities of the categories are given in Table A.3.

Table A.3 PMF of Equal Probability Cases, where $m = 4$

Cases	**Category-1**	**Category-2**	**Category-3**	**Category-4**
Case-1	0.33	0.33	0.33	0
Case-2	0.33	0.33	0	0.33
Case-3	0.33	0	0.33	0.33
Case-4	0	0.33	0.33	0.33

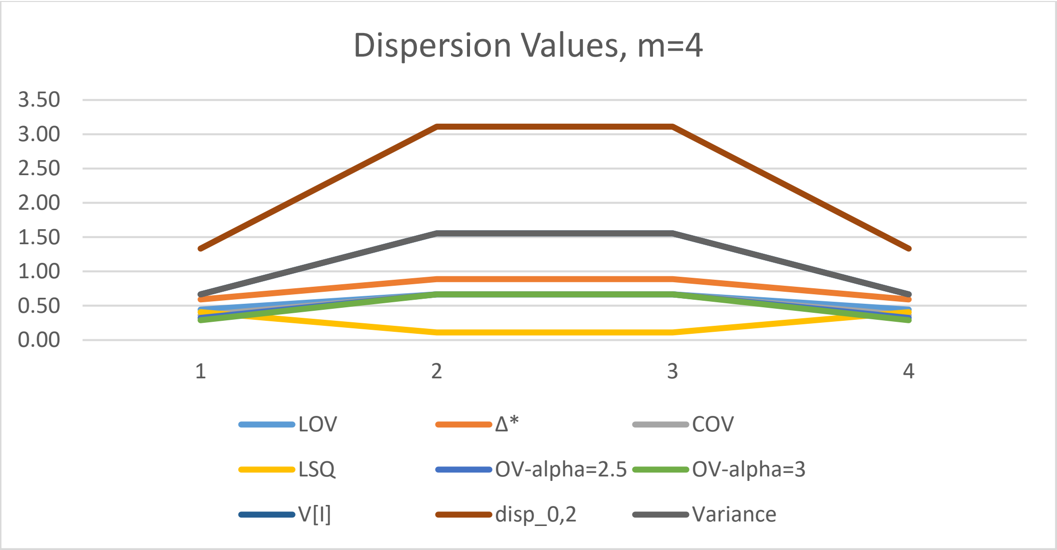


Figure A.3 Dispersion Values Obtained with the Probability Distributions in Table A.2

Table A.4 Dispersion Values Obtained with the Probability Distributions in Table A.2

Cases	**LOV**	Δ^*	**COV**	**LSQ**	$\boldsymbol{OV_{2.5}}$	$\boldsymbol{OV_3}$	**V[I]**	$\boldsymbol{disp_{0,2}}$
Case-1	0.44	0.59	0.36	0.41	0.32	0.29	0.67	1.33
Case-2	0.67	0.89	0.67	0.11	0.67	0.67	0.53	3.11
Case-3	0.67	0.89	0.67	0.11	0.67	0.67	0.50	3.11
Case-4	0.44	0.59	0.36	0.41	0.32	0.29	0.25	1.33

As seen in Figure A.3, Case-1 and Case-4 have equal dispersion and similarly, Case-2 and Case-3 have equal dispersion. Variance and $V(I)$ produced the same dispersion. Location values of the probability distributions given in Table A.2 are provided in Figure A.4.

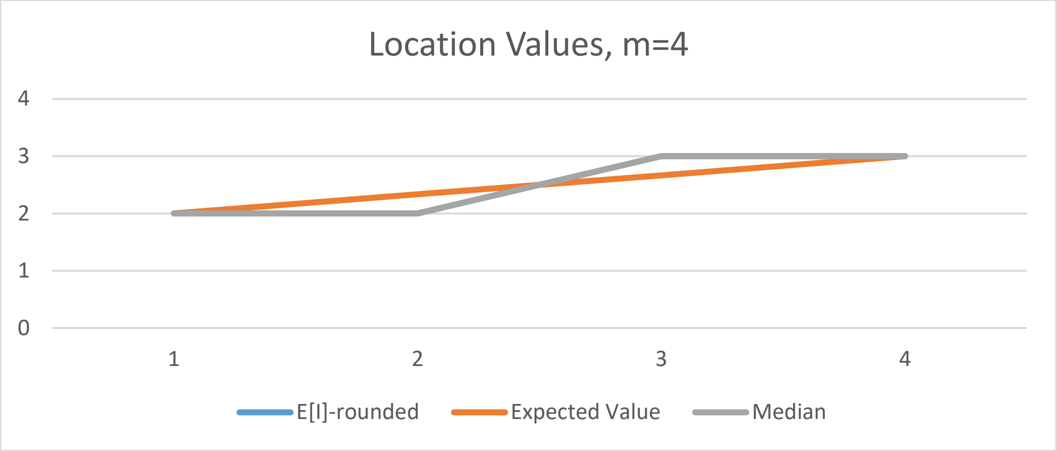


Figure A.4 Location Values Obtained with the Probability Distributions in Table A.2

While rounded-$E[I]$ and median give the same location values for the probability distributions, the expected value increases linearly.

In the final case, equal probabilities for two categories, where $p_i = p_j = 0.5,\ i = 1,2,\dots,m\ and\ j = 1,2,\dots,m$ is examined. Probability distributions are presented in Table A.5.

Table A.5 PMF of Equal Probabilities for $p_i = p_j = 0.5$, where $m = 4$

Cases	**Category-1**	**Category-2**	**Category-3**	**Category-4**
Case-1	0.5	0	0	0.5
Case-2	0.5	0	0.5	0

Table A.5 (cont'd) PMF of Equal Probabilities for $p_i = p_j = 0.5$, where $m = 4$

Case-3	0.5	0.5	0	0
Case-4	0	0.5	0.5	0
Case-5	0	0	0.5	0.5
Case-6	0	0.5	0	0.5

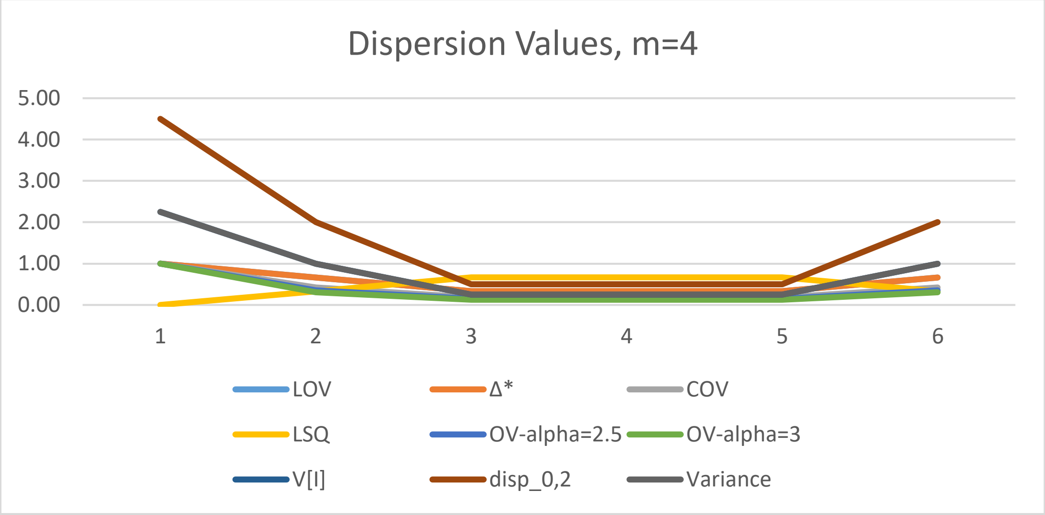


Figure A.5 Dispersion Values Obtained with the Probability Distributions in Table A.3

Table A.6 Dispersion Values Obtained with the Probability Distributions in Table A.3

Cases	LOV	Δ^*	COV	LSQ	$OV_{2.5}$	OV_3	V[I]	$disp_{0,2}$
Case-1	1.00	1.00	1.00	0.00	1.00	1.00	0.56	4.50
Case-2	0.67	0.67	0.42	0.33	0.36	0.31	0.75	2.00
Case-3	0.33	0.33	0.18	0.67	0.15	0.13	0.69	0.50
Case-4	0.33	0.33	0.18	0.67	0.15	0.13	0.44	0.50

Table A.6 (cont'd) Dispersion Values Obtained with the Probability Distributions in Table A.3

Case-5	0.33	0.33	0.18	0.67	0.15	0.13	0.06	0.50
Case-6	0.67	0.67	0.42	0.33	0.36	0.31	0.25	2.00

The dispersion values for the probability distributions given in Table A.5 are provided in Figure A.5. Case-1 shows the extreme two-point distribution. In this case, the maximum dispersion is obtained for all measures. As the categories with probability 0.5 move away from each other, dispersion value increases. Dispersion of Case-2 and Case-6 are equal. Similarly, the dispersion obtained from Case-3, Case-4, and Case-5 are equal with all of the considered dispersion measures.

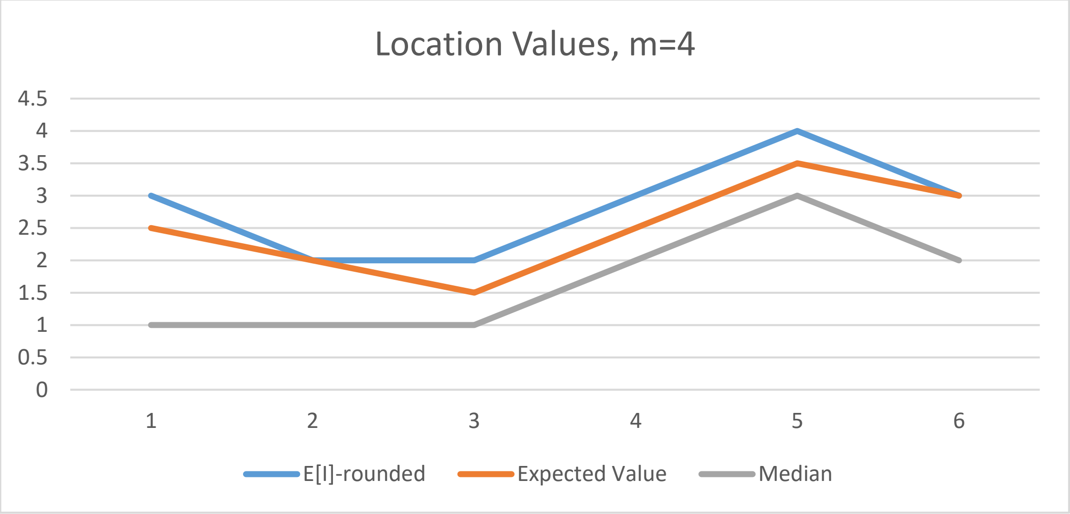


Figure A.6 Location Values Obtained with the Probability Distributions Given in Table A.3

Location values are provided in Figure A.6. While the rounded-$E[I]$ shows a similar distribution to the expected value, the median tends to produce smaller results than these measures.

In the literature, the relationship between measures has been mentioned. These relationships are examined with the Pearson correlation coefficient, ρ, in this study. For variables X and Y Pearson correlation coefficient as a measure of linear relationship is calculated as given in Equation (A.1).

$$\rho = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \times \sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{A.1}$$

where, $\bar{x}$ and $\bar{y}$ are the mean of x and y, respectively. When, $\rho = \pm 1$, then there is a strong linear relationship between two variables. There is strong correlation between two variables, if $\rho \geq \pm 0.5$. It can be said that moderate correlation exists between X and Y, if $\pm 0.3 \leq \rho \leq \pm 0.49$. There is low correlation, if $\rho \leq \pm 0.29$, and when $\rho = 0$, variables are not linearly dependent (Laerd Statistics, 2020).

In order to examine the correlation of location and dispersion measures, the dataset consisting of 200 observations are created according to rank counts I, $I \sim Bin(3, p)$, where p is randomly generated in range $[0, 1]$. Resulting correlations are provided in Figures A.7 and A.8, for dispersion measures and location measures, respectively.

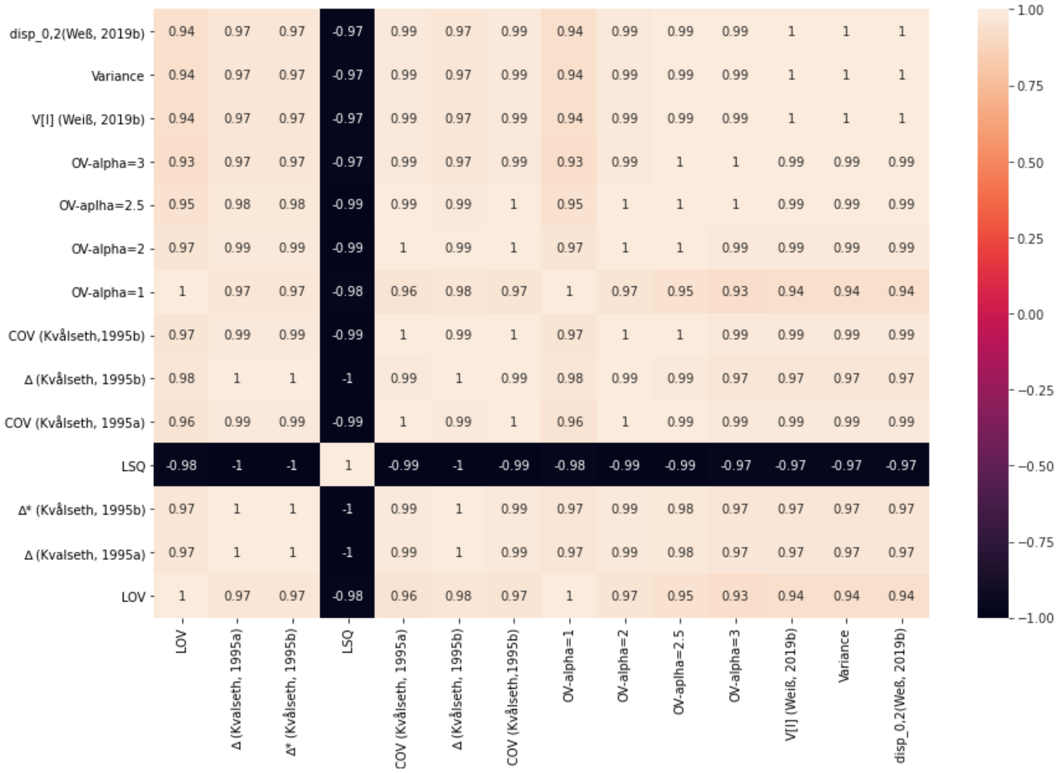


Figure A.7 Correlation Heatmap of Dispersion Measures

All ρ values are greater than 0.90 according to Figure A.7. Therefore, all dispersion measures have a strong correlation with each other. Since $LSQ = 1 - \Delta^*$, these two measures have a perfect negative relationship. Also, since $OV_1 = LOV$ corresponding ρ equals to 1 for these two measures. Similarly, there is a perfect positive relationship between OV_2 and COV, as well. $OV_{2.5}$ has perfect linear relationship between OV_2 and OV_3. Moreover, $V(I)$, variance and $disp_{O,2}$ are highly correlated with each other.

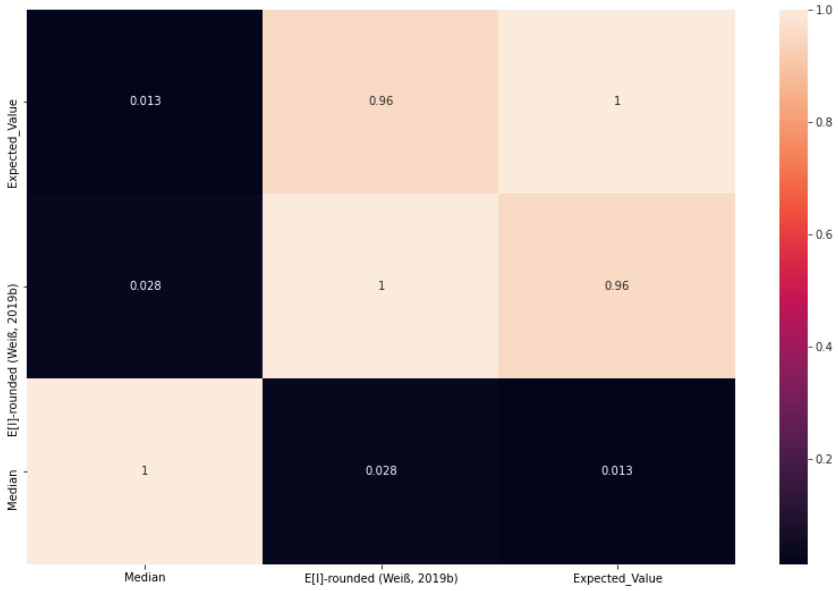


Figure A.8 Correlation Heatmap of Location Measures

The correlation between rounded-$E[I]$ expected value and median are presented in Figure A.8. It is observed in this study that the $E[I]$ gives the same results as the expected value. Therefore, there is a high correlation between them. The median does not have a linear relationship with these measures.

B. CROSS-VALIDATION RESULTS

Table B.1 (cont'd) Repeated Stratified 3-fold Cross-Validation Results for RF, LR and XGB

DataSet	Method	Metric	$\boldsymbol{Fold_{11}}$	$\boldsymbol{Fold_{12}}$	$\boldsymbol{Fol_{d13}}$	$\boldsymbol{Fold_{21}}$	$\boldsymbol{Fold_{22}}$	$\boldsymbol{Fold_{23}}$	$\boldsymbol{Fold_{31}}$	$\boldsymbol{Fold_{32}}$	$\boldsymbol{Fold_{33}}$	Avg
Surface Defect	RF	AUC_{Train}	0.89	0.89	0.90	0.88	0.89	0.91	0.89	0.89	0.90	0.90
		AUC_{Test}	0.79	0.80	0.79	0.81	0.80	0.73	0.80	0.83	0.75	0.79
		$Precision_{Train}$	0.62	0.62	0.60	0.58	0.59	0.62	0.60	0.63	0.62	0.61
		$Precision_{Test}$	0.48	0.51	0.42	0.46	0.57	0.46	0.40	0.50	0.45	0.47
		$Recall_{Train}$	0.61	0.59	0.60	0.57	0.59	0.64	0.60	0.58	0.62	0.60
		$Recall_{Test}$	0.46	0.50	0.41	0.46	0.52	0.45	0.40	0.51	0.44	0.46
	LR	AUC_{Train}	0.83	0.82	0.86	0.84	0.82	0.85	0.83	0.84	0.85	0.84
		AUC_{Test}	0.78	0.83	0.74	0.78	0.83	0.76	0.78	0.79	0.75	0.78

Table B.1 (cont'd) Repeated Stratified 3-fold Cross-Validation Results for RF, LR and XGB

		$Precision_{Train}$	0.61	0.54	0.59	0.56	0.52	0.58	0.54	0.57	0.57	0.56
		$Precision_{Test}$	0.51	0.52	0.44	0.48	0.57	0.43	0.42	0.45	0.35	0.46
		$Recall_{Train}$	0.61	0.53	0.59	0.56	0.51	0.56	0.53	0.54	0.55	0.55
		$Recall_{Test}$	0.48	0.50	0.42	0.45	0.59	0.38	0.48	0.49	0.34	0.46
	XGB	AUC_{Train}	0.89	0.88	0.90	0.88	0.89	0.90	0.89	0.88	0.90	0.89
		AUC_{Test}	0.79	0.83	0.77	0.83	0.81	0.75	0.79	0.83	0.77	0.80
		$Precision_{Train}$	0.65	0.60	0.61	0.55	0.48	0.67	0.62	0.61	0.61	0.60
		$Precision_{Test}$	0.51	0.49	0.45	0.48	0.44	0.49	0.44	0.59	0.44	0.48
		$Recall_{Train}$	0.59	0.57	0.58	0.56	0.55	0.58	0.57	0.53	0.60	0.57
		$Recall_{Test}$	0.45	0.46	0.45	0.46	0.48	0.45	0.42	0.50	0.44	0.46
Inkjet Printer	RF	AUC_{Train}	0.83	0.82	0.88	0.84	0.86	0.81	0.79	0.92	0.81	0.84
		AUC_{Test}	0.72	0.70	0.63	0.72	0.69	0.79	0.81	0.52	0.79	0.71

Table B.1 (cont'd) Repeated Stratified 3-fold Cross-Validation Results for RF, LR and XGB

		$Precision_{Train}$	0.55	0.49	0.63	0.48	0.59	0.51	0.53	0.72	0.53	0.56
		$Precision_{Test}$	0.50	0.43	0.40	0.45	0.44	0.64	0.55	0.26	0.56	0.47
		$Recall_{Train}$	0.56	0.58	0.67	0.60	0.61	0.53	0.53	0.71	0.55	0.59
		$Recall_{Test}$	0.47	0.50	0.35	0.50	0.53	0.68	0.55	0.24	0.54	0.48
	LR	AUC_{Train}	0.75	0.78	0.83	0.78	0.84	0.73	0.71	0.89	0.76	0.79
		AUC_{Test}	0.76	0.59	0.61	0.71	0.64	0.77	0.82	0.52	0.70	0.68
		$Precision_{Train}$	0.34	0.30	0.47	0.39	0.45	0.23	0.34	0.55	0.29	0.37
		$Precision_{Test}$	0.46	0.28	0.26	0.38	0.28	0.22	0.46	0.28	0.25	0.32
		$Recall_{Train}$	0.34	0.42	0.42	0.41	0.41	0.30	0.34	0.63	0.33	0.40
		$Recall_{Test}$	0.44	0.38	0.29	0.32	0.32	0.31	0.44	0.32	0.30	0.34
	XGB	AUC_{Train}	0.83	0.82	0.88	0.84	0.86	0.82	0.79	0.92	0.81	0.84
		AUC_{Test}	0.75	0.69	0.69	0.75	0.71	0.83	0.65	0.60	0.81	0.72

Table B.1 (cont'd) Repeated Stratified 3-fold Cross-Validation Results for RF, LR and XGB

		$Precision_{Train}$	0.44	0.46	0.62	0.47	0.47	0.43	0.37	0.55	0.41	0.47
		$Precision_{Test}$	0.43	0.39	0.28	0.38	0.45	0.39	0.63	0.28	0.43	0.41
		$Recall_{Train}$	0.52	0.50	0.53	0.54	0.53	0.46	0.43	0.63	0.46	0.51
		$Recall_{Test}$	0.49	0.44	0.27	0.39	0.55	0.44	0.58	0.32	0.45	0.44
Duplicator	RF	AUC_{Train}	0.72	0.81	0.84	0.78	0.80	0.76	0.72	0.79	0.75	0.77
		AUC_{Test}	0.63	0.64	0.64	0.58	0.58	0.62	0.65	0.73	0.71	0.64
		$Precision_{Train}$	0.45	0.54	0.33	0.36	0.35	0.48	0.38	0.49	0.49	0.43
		$Precision_{Test}$	0.26	0.47	0.28	0.23	0.26	0.38	0.25	0.29	0.36	0.31
		$Recall_{Train}$	0.44	0.54	0.49	0.47	0.45	0.41	0.52	0.53	0.47	0.48
		$Recall_{Test}$	0.31	0.42	0.31	0.29	0.27	0.25	0.31	0.40	0.37	0.33
	LR	AUC_{Train}	0.70	0.78	0.70	0.72	0.71	0.78	0.69	0.80	0.78	0.74
		AUC_{Test}	0.55	0.67	0.48	0.57	0.62	0.62	0.55	0.70	0.64	0.60

Table B.1 (cont'd) Repeated Stratified 3-fold Cross-Validation Results for RF, LR and XGB

		$Precision_{Train}$	0.45	0.45	0.48	0.51	0.44	0.52	0.51	0.37	0.44	0.46
		$Precision_{Test}$	0.26	0.42	0.25	0.23	0.21	0.48	0.21	0.31	0.34	0.30
		$Recall_{Train}$	0.43	0.38	0.46	0.51	0.41	0.30	0.40	0.40	0.41	0.41
		$Recall_{Test}$	0.33	0.39	0.16	0.19	0.21	0.32	0.19	0.31	0.21	0.26
	XGB	AUC_{Train}	0.78	0.76	0.67	0.59	0.72	0.70	0.68	0.74	0.85	0.72
		AUC_{Test}	0.58	0.73	0.57	0.51	0.60	0.57	0.64	0.63	0.77	0.62
		$Precision_{Train}$	0.37	0.36	0.41	0.41	0.35	0.36	0.33	0.37	0.40	0.37
		$Precision_{Test}$	0.27	0.40	0.21	0.32	0.26	0.41	0.31	0.39	0.36	0.33
		$Recall_{Train}$	0.43	0.36	0.44	0.38	0.43	0.39	0.42	0.39	0.36	0.40
		$Recall_{Test}$	0.40	0.40	0.26	0.38	0.33	0.35	0.35	0.35	0.40	0.36
Foam Molding	RF	AUC_{Train}	0.73	0.74	0.72	0.72	0.72	0.74	0.74	0.72	0.74	0.73
		AUC_{Test}	0.66	0.67	0.72	0.70	0.71	0.64	0.69	0.72	0.65	0.69

Table B.1 (cont'd) Repeated Stratified 3-fold Cross-Validation Results for RF, LR and XGB

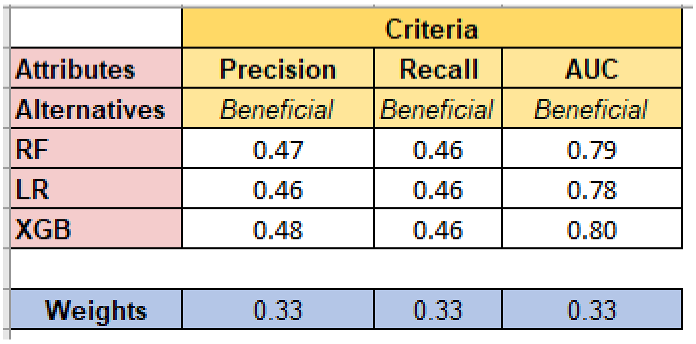
		$Precision_{Train}$	0.48	0.51	0.50	0.28	0.50	0.54	0.52	0.48	0.53	0.48
		$Precision_{Test}$	0.45	0.44	0.47	0.26	0.43	0.44	0.43	0.48	0.46	0.43
		$Recall_{Train}$	0.57	0.57	0.53	0.57	0.56	0.57	0.58	0.54	0.57	0.56
		$Recall_{Test}$	0.51	0.47	0.52	0.51	0.51	0.49	0.47	0.52	0.38	0.49
	LR	AUC_{Train}	0.73	0.74	0.72	0.71	0.72	0.74	0.74	0.72	0.74	0.73
		AUC_{Test}	0.65	0.67	0.72	0.68	0.71	0.66	0.67	0.72	0.64	0.68
		$Precision_{Train}$	0.48	0.51	0.49	0.28	0.50	0.54	0.50	0.48	0.51	0.48
		$Precision_{Test}$	0.43	0.42	0.44	0.26	0.43	0.44	0.44	0.48	0.40	0.41
		$Recall_{Train}$	0.57	0.57	0.54	0.57	0.56	0.57	0.58	0.54	0.57	0.56
		$Recall_{Test}$	0.51	0.47	0.54	0.51	0.51	0.46	0.46	0.52	0.38	0.48
	XGB	AUC_{Train}	0.74	0.76	0.73	0.75	0.73	0.75	0.75	0.73	0.75	0.74
		AUC_{Test}	0.68	0.68	0.70	0.67	0.72	0.67	0.69	0.69	0.66	0.68

Table B.1 (cont'd) Repeated Stratified 3-fold Cross-Validation Results for RF, LR and XGB

		$Precision_{Train}$	0.44	0.43	0.44	0.41	0.44	0.45	0.44	0.44	0.52	0.44
		$Precision_{Test}$	0.42	0.36	0.41	0.40	0.45	0.42	0.45	0.43	0.49	0.42
		$Recall_{Train}$	0.45	0.47	0.44	0.45	0.45	0.47	0.45	0.45	0.44	0.45
		$Recall_{Test}$	0.45	0.41	0.47	0.46	0.45	0.42	0.46	0.45	0.41	0.44

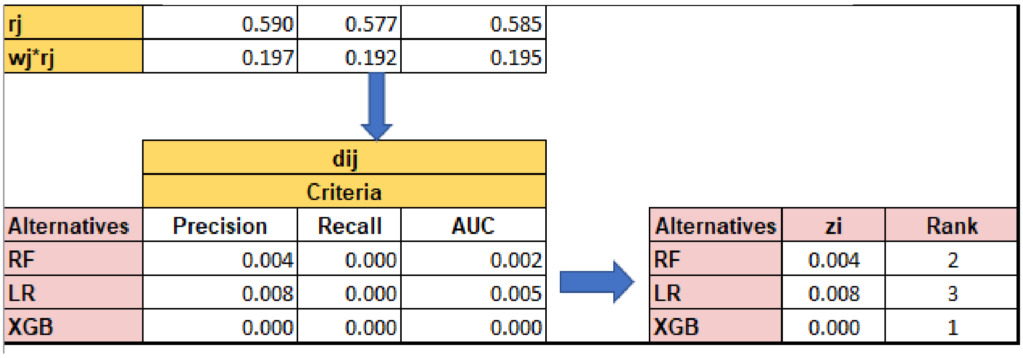
C. SURFACE DEFECT CASE TOPSIS AND MULTI-MOORA RESULTS

Multi-MOORA with Equal Weights:



	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.47	0.46	0.79
LR	0.46	0.46	0.78
XGB	0.48	0.46	0.80
Weights	0.33	0.33	0.33

Figure C.1 Decision Matrix

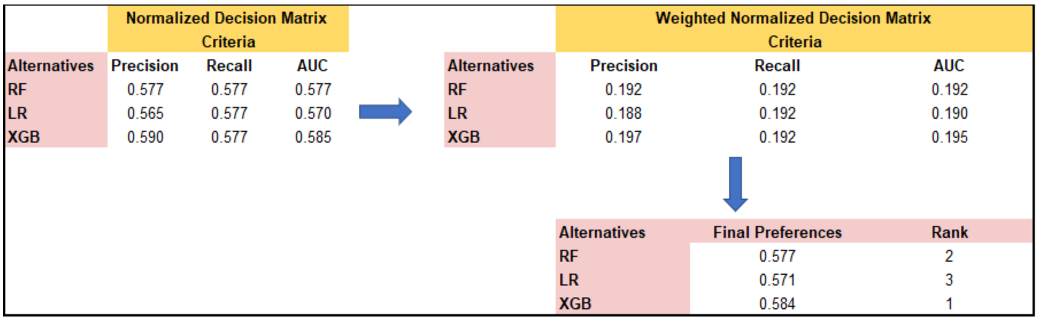


rj	0.590	0.577	0.585
wj*rj	0.197	0.192	0.195

	dij		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.004	0.000	0.002
LR	0.008	0.000	0.005
XGB	0.000	0.000	0.000

Alternatives	zi	Rank
RF	0.004	2
LR	0.008	3
XGB	0.000	1

Figure C.2 Ratio System

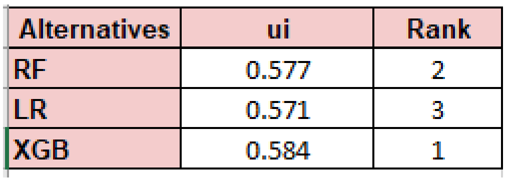


	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.577	0.577	0.577
LR	0.565	0.577	0.570
XGB	0.590	0.577	0.585

	Weighted Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.192	0.192	0.192
LR	0.188	0.192	0.190
XGB	0.197	0.192	0.195

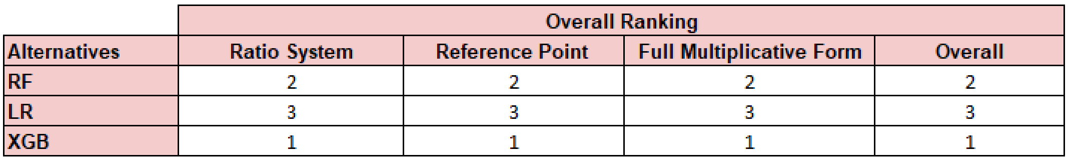
Alternatives	Final Preferences	Rank
RF	0.577	2
LR	0.571	3
XGB	0.584	1

Figure C.3 Reference Point Approach



Alternatives	ui	Rank
RF	0.577	2
LR	0.571	3
XGB	0.584	1

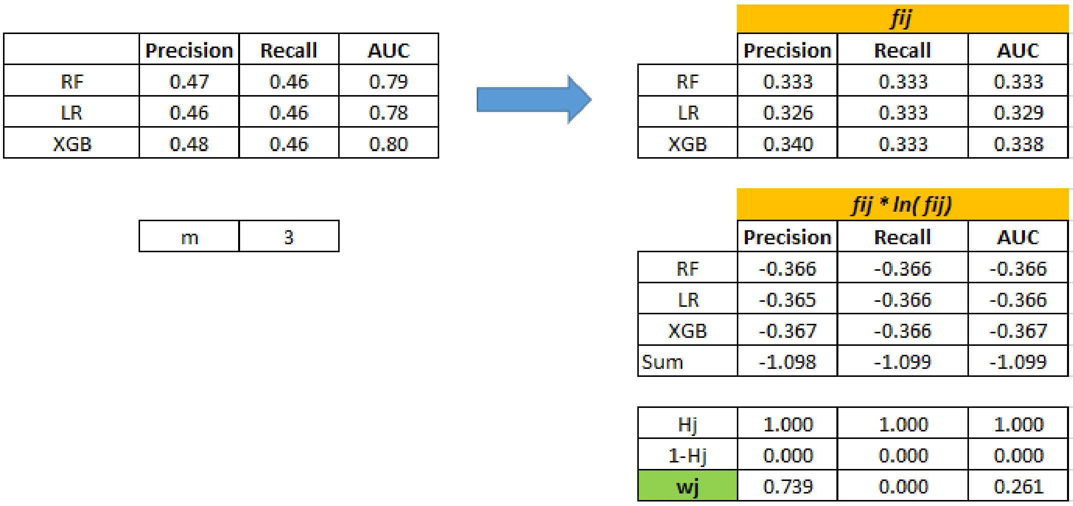
Figure C.4 Full Multiplicative Form



	Overall Ranking			
Alternatives	Ratio System	Reference Point	Full Multiplicative Form	Overall
RF	2	2	2	2
LR	3	3	3	3
XGB	1	1	1	1

Figure C.5 Overall Results of Multi-MOORA

Multi-MOORA with Entropy Weights:



	Precision	Recall	AUC
RF	0.47	0.46	0.79
LR	0.46	0.46	0.78
XGB	0.48	0.46	0.80

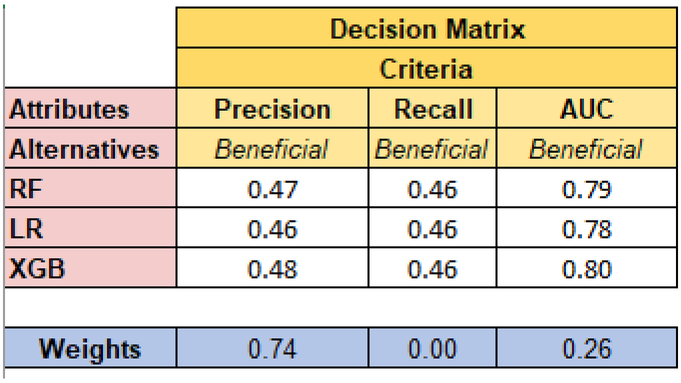
m	3

	fij		
	Precision	Recall	AUC
RF	0.333	0.333	0.333
LR	0.326	0.333	0.329
XGB	0.340	0.333	0.338

	*fij * ln(fij)*		
	Precision	Recall	AUC
RF	-0.366	-0.366	-0.366
LR	-0.365	-0.366	-0.366
XGB	-0.367	-0.366	-0.367
Sum	-1.098	-1.099	-1.099

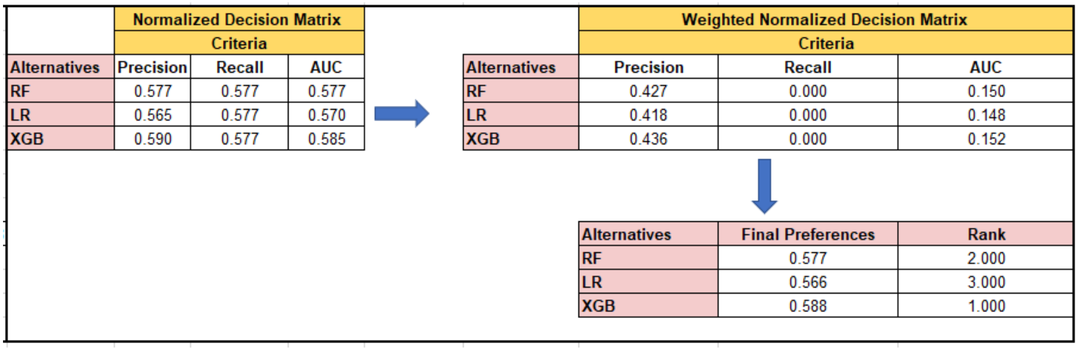
Hj	1.000	1.000	1.000
1-Hj	0.000	0.000	0.000
wj	0.739	0.000	0.261

Figure C.6 Calculation of Entropy Weights



	Decision Matrix		
	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.47	0.46	0.79
LR	0.46	0.46	0.78
XGB	0.48	0.46	0.80
Weights	0.74	0.00	0.26

Figure C.7 Decision Matrix

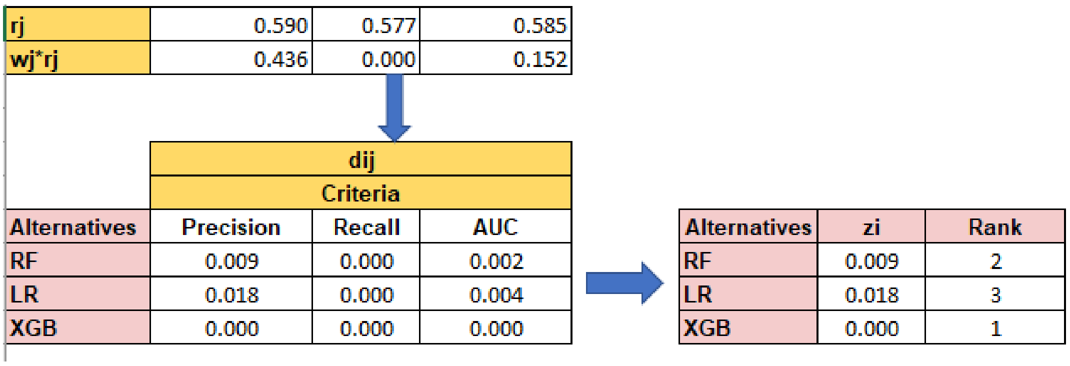


	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.577	0.577	0.577
LR	0.565	0.577	0.570
XGB	0.590	0.577	0.585

	Weighted Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.427	0.000	0.150
LR	0.418	0.000	0.148
XGB	0.436	0.000	0.152

Alternatives	Final Preferences	Rank
RF	0.577	2.000
LR	0.566	3.000
XGB	0.588	1.000

Figure C.8 Ratio System

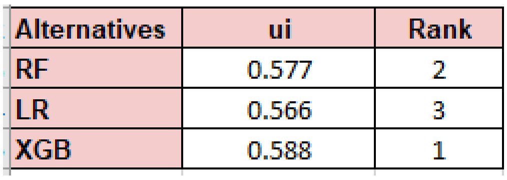


rj	0.590	0.577	0.585
wj*rj	0.436	0.000	0.152

	dij		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.009	0.000	0.002
LR	0.018	0.000	0.004
XGB	0.000	0.000	0.000

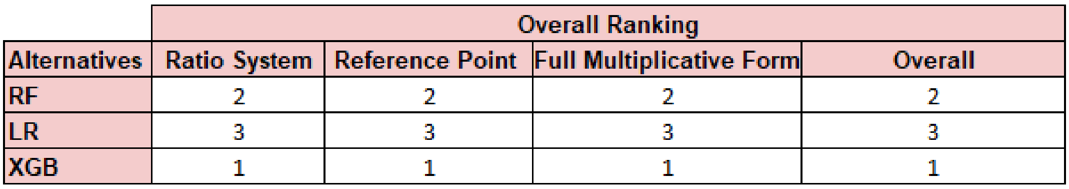
Alternatives	zi	Rank
RF	0.009	2
LR	0.018	3
XGB	0.000	1

Figure C.9 Reference Point Approach



Alternatives	ui	Rank
RF	0.577	2
LR	0.566	3
XGB	0.588	1

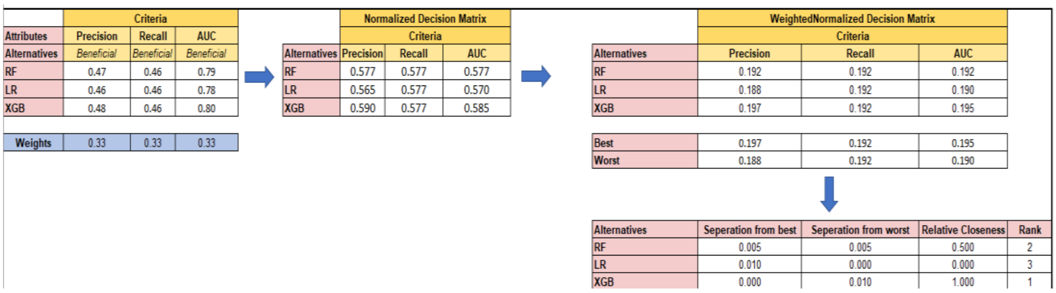
Figure C.10 Full Multiplicative Form



	Overall Ranking			
Alternatives	Ratio System	Reference Point	Full Multiplicative Form	Overall
RF	2	2	2	2
LR	3	3	3	3
XGB	1	1	1	1

Figure C.11 Overall Results of Multi-MOORA

TOPSIS with Equal Weights:



	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.47	0.46	0.79
LR	0.46	0.46	0.78
XGB	0.48	0.46	0.80
Weights	0.33	0.33	0.33

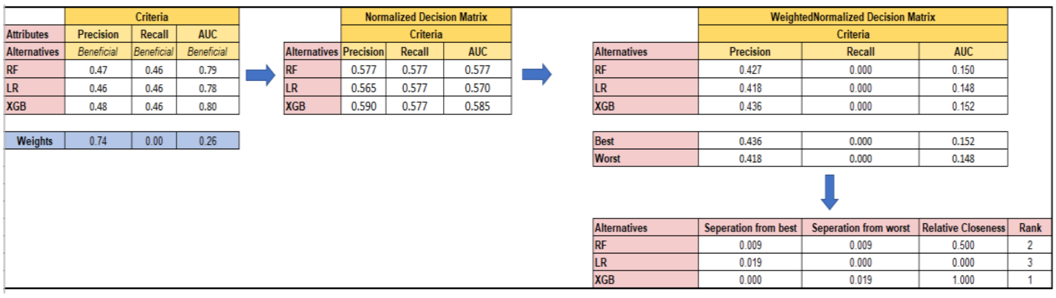
	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.577	0.577	0.577
LR	0.565	0.577	0.570
XGB	0.590	0.577	0.585

	WeightedNormalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.192	0.192	0.192
LR	0.188	0.192	0.190
XGB	0.197	0.192	0.195
Best	0.197	0.192	0.195
Worst	0.188	0.192	0.190

Alternatives	Seperation from best	Seperation from worst	Relative Closeness	Rank
RF	0.005	0.005	0.500	2
LR	0.010	0.000	0.000	3
XGB	0.000	0.010	1.000	1

Figure C.12 TOPSIS with Equal Weights

TOPSIS with Entropy Weights:



	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.47	0.46	0.79
LR	0.46	0.46	0.78
XGB	0.48	0.46	0.80
Weights	0.74	0.00	0.26

	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.577	0.577	0.577
LR	0.565	0.577	0.570
XGB	0.590	0.577	0.585

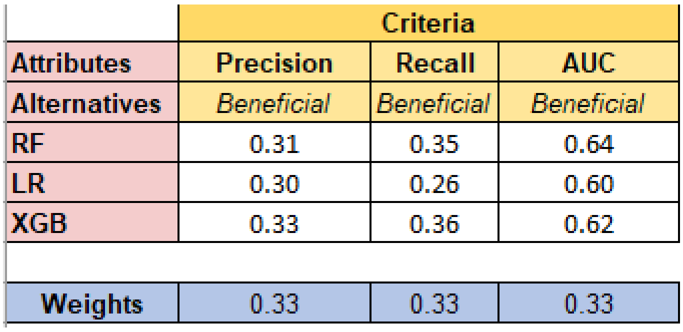
	WeightedNormalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.427	0.000	0.150
LR	0.418	0.000	0.148
XGB	0.436	0.000	0.152
Best	0.436	0.000	0.152
Worst	0.418	0.000	0.148

Alternatives	Seperation from best	Seperation from worst	Relative Closeness	Rank
RF	0.009	0.009	0.500	2
LR	0.019	0.000	0.000	3
XGB	0.000	0.019	1.000	1

Figure C.13 TOPSIS with Entropy Weights

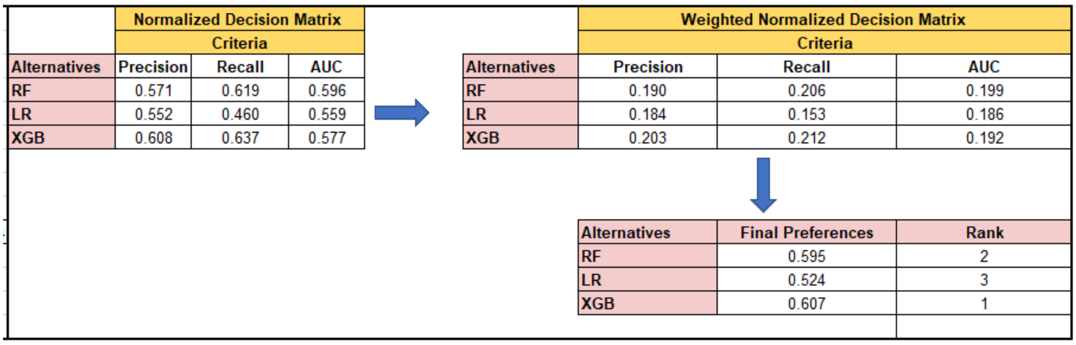
D. DUPLICATOR CASE TOPSIS AND MULTI-MOORA RESULT

Multi-MOORA with Equal Weights:



	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.31	0.35	0.64
LR	0.30	0.26	0.60
XGB	0.33	0.36	0.62
Weights	0.33	0.33	0.33

Figure D.1 Decision Matrix

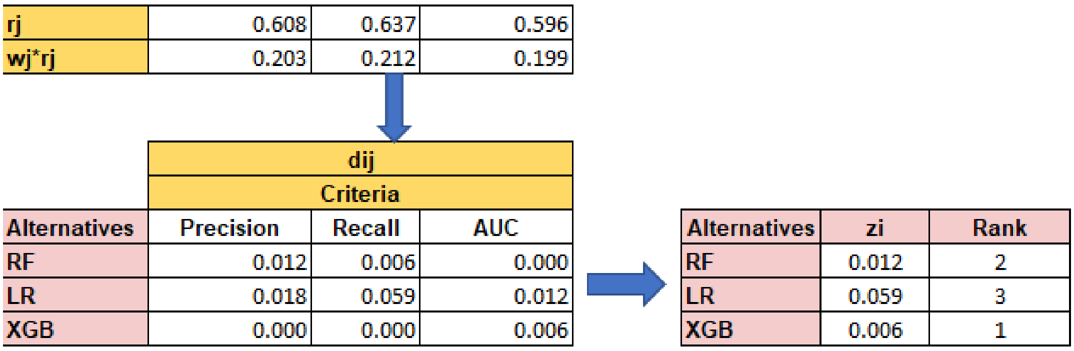


	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.571	0.619	0.596
LR	0.552	0.460	0.559
XGB	0.608	0.637	0.577

	Weighted Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.190	0.206	0.199
LR	0.184	0.153	0.186
XGB	0.203	0.212	0.192

Alternatives	Final Preferences	Rank
RF	0.595	2
LR	0.524	3
XGB	0.607	1

Figure D.2 Ratio System

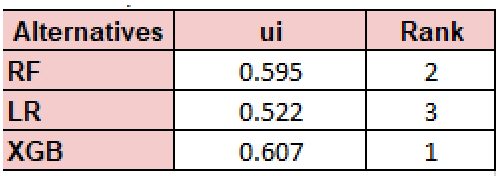


rj	0.608	0.637	0.596
wj*rj	0.203	0.212	0.199

	dij		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.012	0.006	0.000
LR	0.018	0.059	0.012
XGB	0.000	0.000	0.006

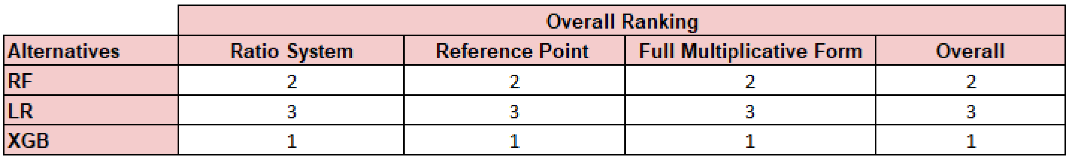
Alternatives	zi	Rank
RF	0.012	2
LR	0.059	3
XGB	0.006	1

Figure D.3 Reference Point Approach



Alternatives	ui	Rank
RF	0.595	2
LR	0.522	3
XGB	0.607	1

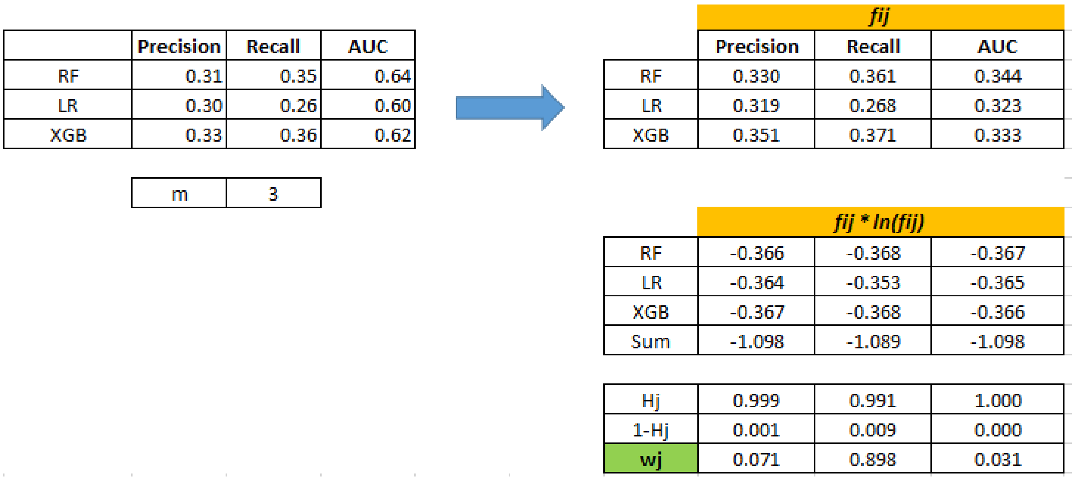
Figure D.4 Full Multiplicative Form



	Overall Ranking			
Alternatives	Ratio System	Reference Point	Full Multiplicative Form	Overall
RF	2	2	2	2
LR	3	3	3	3
XGB	1	1	1	1

Figure D.5 Overall Results of Multi-MOORA

<u>Multi-MOORA with Entropy Weights:</u>



	Precision	Recall	AUC
RF	0.31	0.35	0.64
LR	0.30	0.26	0.60
XGB	0.33	0.36	0.62

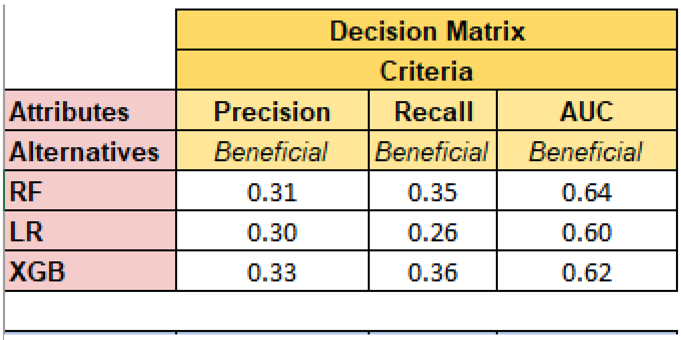
m	3

	fij		
	Precision	Recall	AUC
RF	0.330	0.361	0.344
LR	0.319	0.268	0.323
XGB	0.351	0.371	0.333

	*fij * ln(fij)*		
RF	-0.366	-0.368	-0.367
LR	-0.364	-0.353	-0.365
XGB	-0.367	-0.368	-0.366
Sum	-1.098	-1.089	-1.098

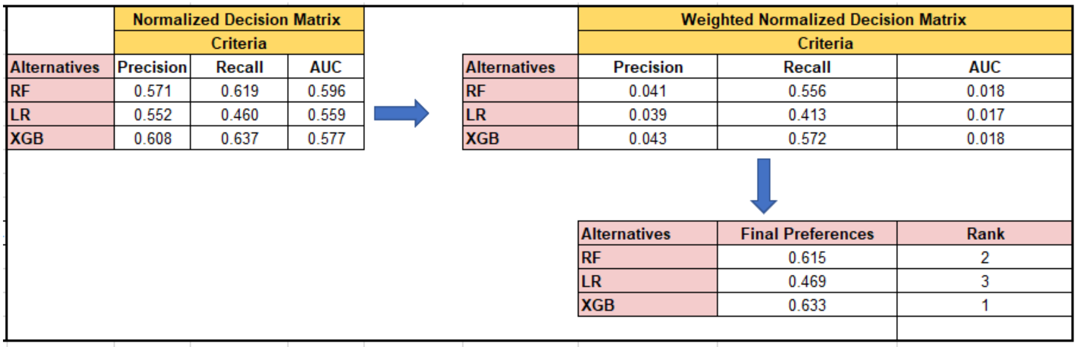
Hj	0.999	0.991	1.000
1-Hj	0.001	0.009	0.000
wj	0.071	0.898	0.031

Figure D.6 Calculation of Entropy Weights



	Decision Matrix		
	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.31	0.35	0.64
LR	0.30	0.26	0.60
XGB	0.33	0.36	0.62

Figure D.7 Decision Matrix

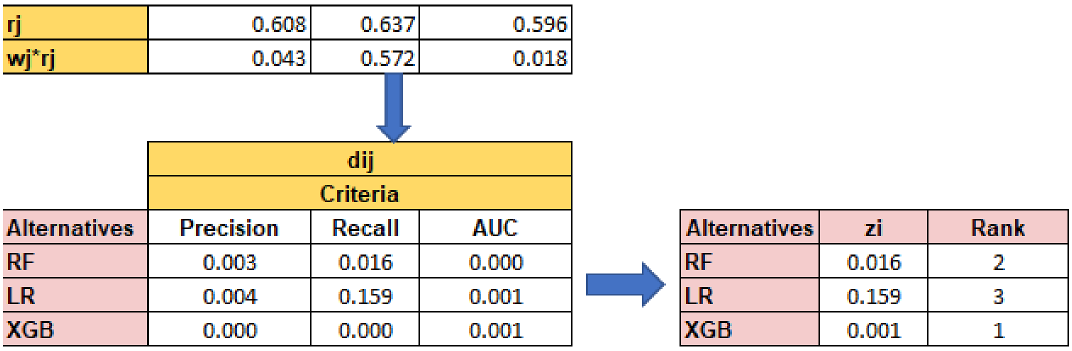


	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.571	0.619	0.596
LR	0.552	0.460	0.559
XGB	0.608	0.637	0.577

	Weighted Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.041	0.556	0.018
LR	0.039	0.413	0.017
XGB	0.043	0.572	0.018

Alternatives	Final Preferences	Rank
RF	0.615	2
LR	0.469	3
XGB	0.633	1

Figure D.8 Ratio System

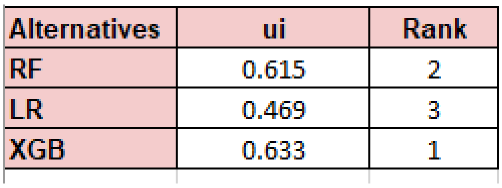


rj	0.608	0.637	0.596
wj*rj	0.043	0.572	0.018

	dij		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.003	0.016	0.000
LR	0.004	0.159	0.001
XGB	0.000	0.000	0.001

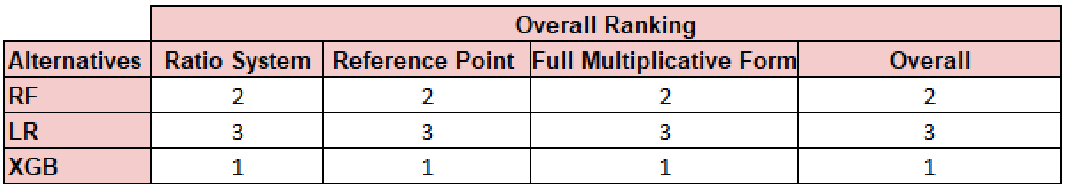
Alternatives	zi	Rank
RF	0.016	2
LR	0.159	3
XGB	0.001	1

Figure D.9 Reference Point Approach



Alternatives	ui	Rank
RF	0.615	2
LR	0.469	3
XGB	0.633	1

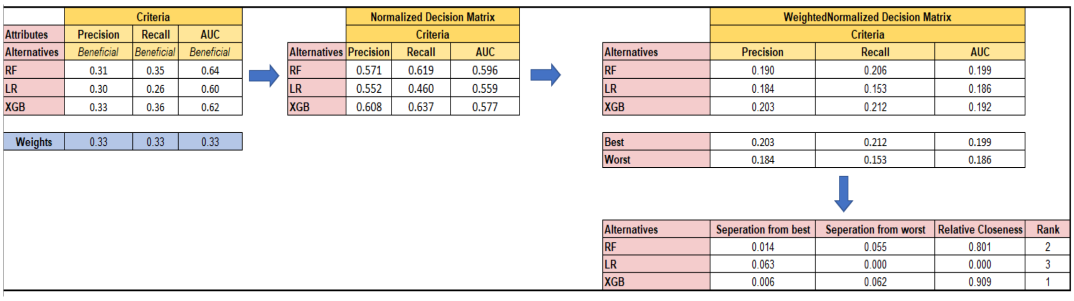
Figure D.10 Full Multiplicative Form



	Overall Ranking			
Alternatives	Ratio System	Reference Point	Full Multiplicative Form	Overall
RF	2	2	2	2
LR	3	3	3	3
XGB	1	1	1	1

Figure D.11 Overall Results of Multi-MOORA

TOPSIS with Equal Weights:



	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.31	0.35	0.64
LR	0.30	0.26	0.60
XGB	0.33	0.36	0.62
Weights	0.33	0.33	0.33

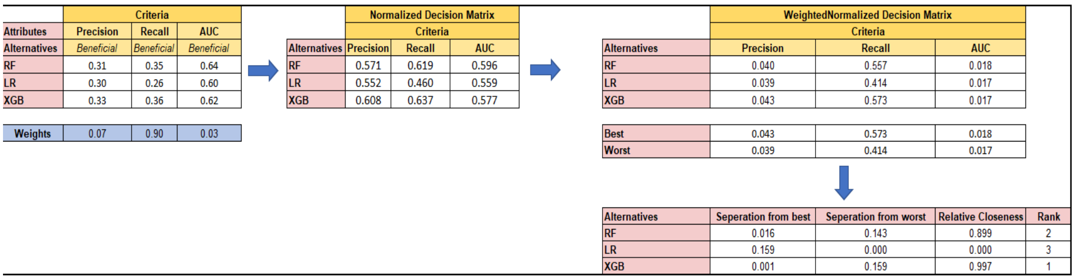
	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.571	0.619	0.596
LR	0.552	0.460	0.559
XGB	0.608	0.637	0.577

	WeightedNormalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.190	0.206	0.199
LR	0.184	0.153	0.186
XGB	0.203	0.212	0.192
Best	0.203	0.212	0.199
Worst	0.184	0.153	0.186

Alternatives	Seperation from best	Seperation from worst	Relative Closeness	Rank
RF	0.014	0.055	0.801	2
LR	0.063	0.000	0.000	3
XGB	0.006	0.062	0.909	1

Figure D.12 TOPSIS with Equal Weights

TOPSIS with Entropy Weights:



	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.31	0.35	0.64
LR	0.30	0.26	0.60
XGB	0.33	0.36	0.62
Weights	0.07	0.90	0.03

	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.571	0.619	0.596
LR	0.552	0.460	0.559
XGB	0.608	0.637	0.577

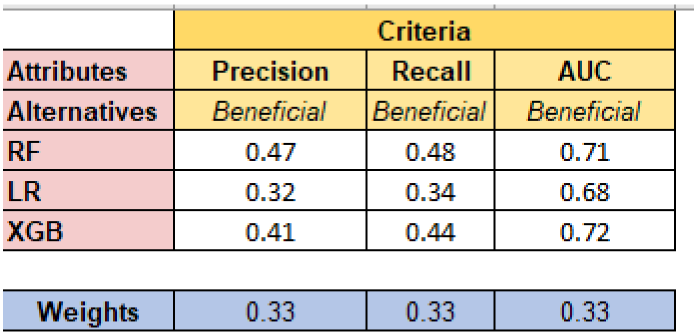
	WeightedNormalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.040	0.557	0.018
LR	0.039	0.414	0.017
XGB	0.043	0.573	0.017
Best	0.043	0.573	0.018
Worst	0.039	0.414	0.017

Alternatives	Seperation from best	Seperation from worst	Relative Closeness	Rank
RF	0.016	0.143	0.899	2
LR	0.159	0.000	0.000	3
XGB	0.001	0.159	0.997	1

Figure D.13 TOPSIS with Entropy Weights

E. INKJET PRINTER CASE TOPSIS AND MULTI-MOORA RESULT

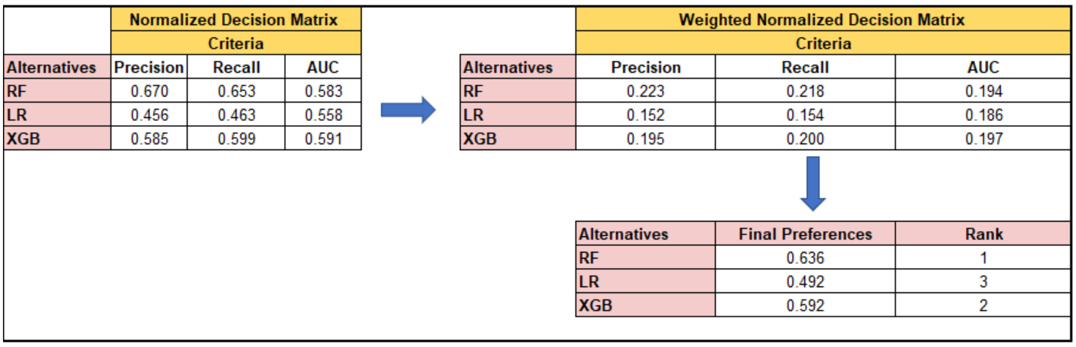
Multi-MOORA with Equal Weights:



	Criteria		
Attributes	**Precision**	**Recall**	**AUC**
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.47	0.48	0.71
LR	0.32	0.34	0.68
XGB	0.41	0.44	0.72

Weights	0.33	0.33	0.33

Figure E.1 Decision Matrix

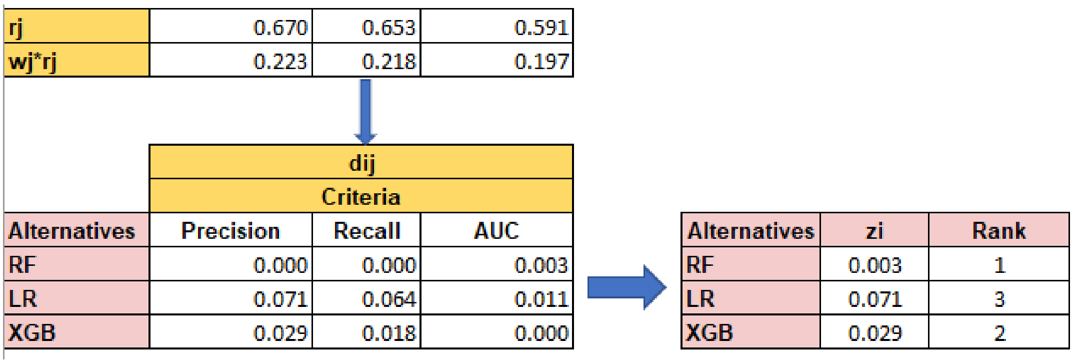


	Normalized Decision Matrix		
	Criteria		
Alternatives	**Precision**	**Recall**	**AUC**
RF	0.670	0.653	0.583
LR	0.456	0.463	0.558
XGB	0.585	0.599	0.591

	Weighted Normalized Decision Matrix		
	Criteria		
Alternatives	**Precision**	**Recall**	**AUC**
RF	0.223	0.218	0.194
LR	0.152	0.154	0.186
XGB	0.195	0.200	0.197

Alternatives	**Final Preferences**	**Rank**
RF	0.636	1
LR	0.492	3
XGB	0.592	2

Figure E.2 Ratio System

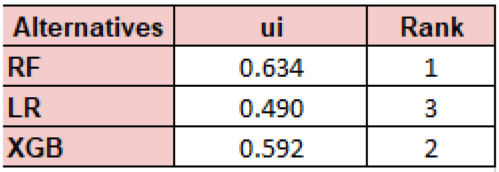


rj	0.670	0.653	0.591
wj*rj	0.223	0.218	0.197

	dij		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.000	0.000	0.003
LR	0.071	0.064	0.011
XGB	0.029	0.018	0.000

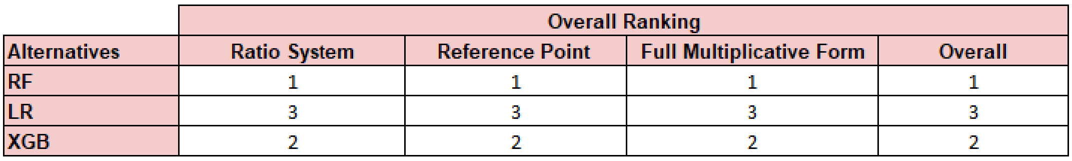
Alternatives	zi	Rank
RF	0.003	1
LR	0.071	3
XGB	0.029	2

Figure E.3 Reference Point Approach



Alternatives	ui	Rank
RF	0.634	1
LR	0.490	3
XGB	0.592	2

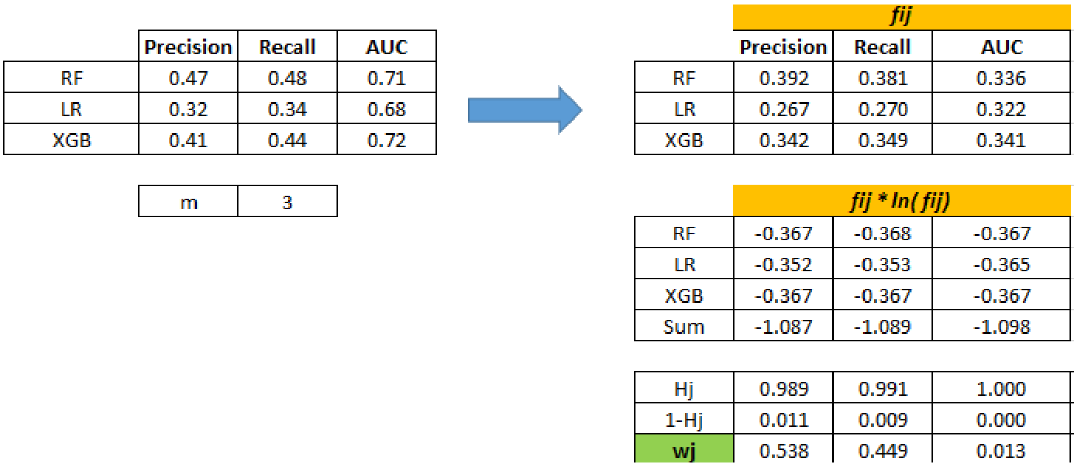
Figure E.4 Full Multiplicative Form



	Overall Ranking			
Alternatives	Ratio System	Reference Point	Full Multiplicative Form	Overall
RF	1	1	1	1
LR	3	3	3	3
XGB	2	2	2	2

Figure E.5 Overall Results of Multi-MOORA

Multi-MOORA with Entropy Weights:



	Precision	Recall	AUC
RF	0.47	0.48	0.71
LR	0.32	0.34	0.68
XGB	0.41	0.44	0.72

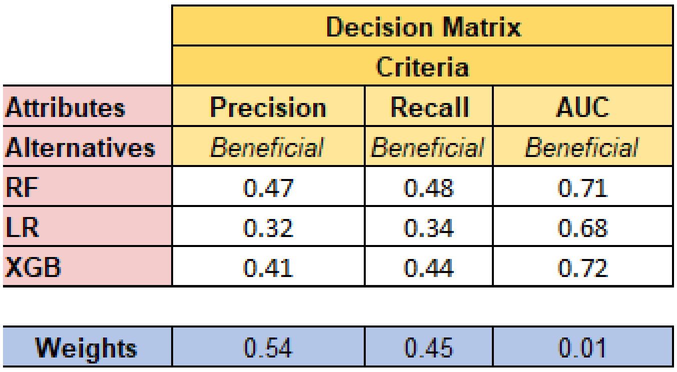
m	3

	fij		
	Precision	Recall	AUC
RF	0.392	0.381	0.336
LR	0.267	0.270	0.322
XGB	0.342	0.349	0.341

	*fij * ln(fij)*		
RF	-0.367	-0.368	-0.367
LR	-0.352	-0.353	-0.365
XGB	-0.367	-0.367	-0.367
Sum	-1.087	-1.089	-1.098

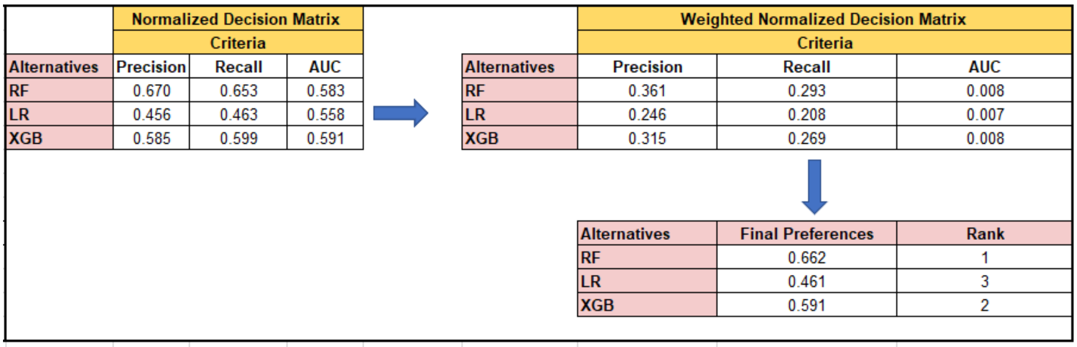
Hj	0.989	0.991	1.000
1-Hj	0.011	0.009	0.000
wj	0.538	0.449	0.013

Figure E.6 Calculation of Entropy Weights



	Decision Matrix		
	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.47	0.48	0.71
LR	0.32	0.34	0.68
XGB	0.41	0.44	0.72
Weights	0.54	0.45	0.01

Figure E.7 Decision Matrix

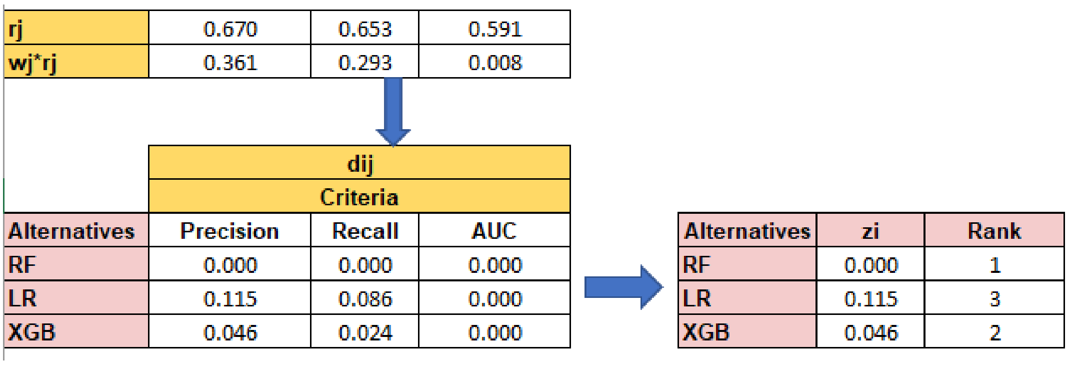


	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.670	0.653	0.583
LR	0.456	0.463	0.558
XGB	0.585	0.599	0.591

	Weighted Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.361	0.293	0.008
LR	0.246	0.208	0.007
XGB	0.315	0.269	0.008

Alternatives	Final Preferences	Rank
RF	0.662	1
LR	0.461	3
XGB	0.591	2

Figure E.8 Ratio System

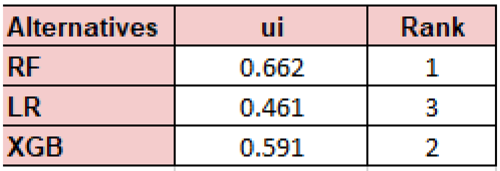


rj	0.670	0.653	0.591
wj*rj	0.361	0.293	0.008

	dij		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.000	0.000	0.000
LR	0.115	0.086	0.000
XGB	0.046	0.024	0.000

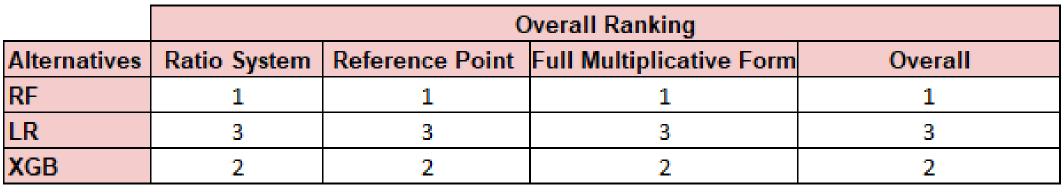
Alternatives	zi	Rank
RF	0.000	1
LR	0.115	3
XGB	0.046	2

Figure E.9 Reference Point Approach



Alternatives	ui	Rank
RF	0.662	1
LR	0.461	3
XGB	0.591	2

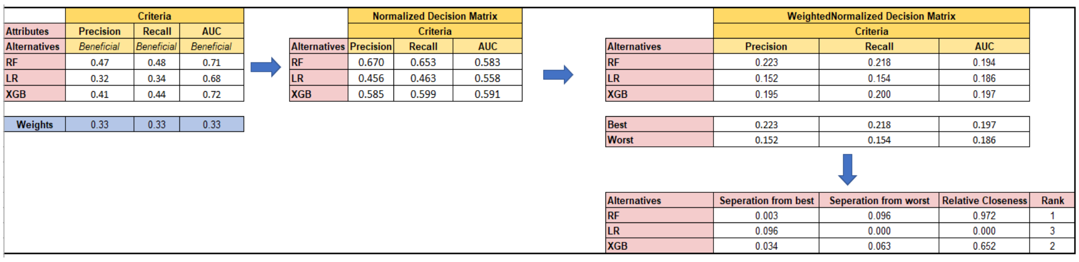
Figure E.10 Full Multiplicative Form



	Overall Ranking			
Alternatives	Ratio System	Reference Point	Full Multiplicative Form	Overall
RF	1	1	1	1
LR	3	3	3	3
XGB	2	2	2	2

Figure E.11 Overall Results of Multi-MOORA

TOPSIS with Equal Weights:



	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.47	0.48	0.71
LR	0.32	0.34	0.68
XGB	0.41	0.44	0.72

Weights	0.33	0.33	0.33

	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.670	0.653	0.583
LR	0.456	0.463	0.558
XGB	0.585	0.599	0.591

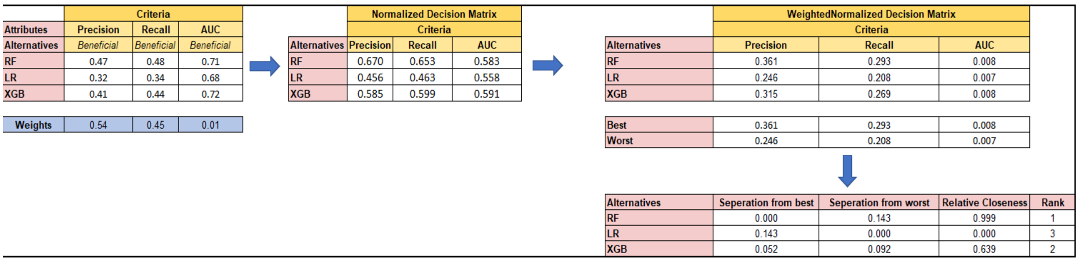
	WeightedNormalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.223	0.218	0.194
LR	0.152	0.154	0.186
XGB	0.195	0.200	0.197

Best	0.223	0.218	0.197
Worst	0.152	0.154	0.186

Alternatives	Seperation from best	Seperation from worst	Relative Closeness	Rank
RF	0.003	0.096	0.972	1
LR	0.096	0.000	0.000	3
XGB	0.034	0.063	0.652	2

Figure E.12 TOPSIS with Equal Weights

TOPSIS with Entropy Weights:



	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.47	0.48	0.71
LR	0.32	0.34	0.68
XGB	0.41	0.44	0.72

Weights	0.54	0.45	0.01

	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.670	0.653	0.583
LR	0.456	0.463	0.558
XGB	0.585	0.599	0.591

	WeightedNormalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.361	0.293	0.008
LR	0.246	0.208	0.007
XGB	0.315	0.269	0.008

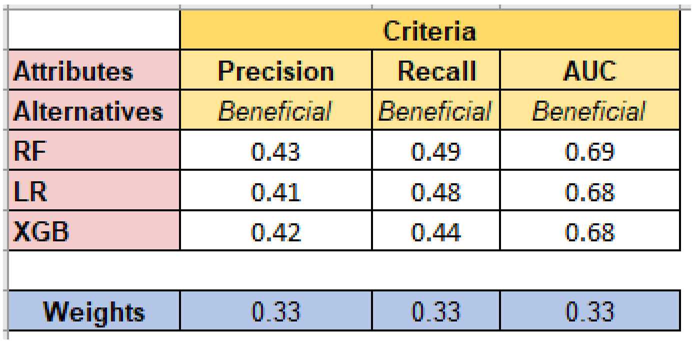
Best	0.361	0.293	0.008
Worst	0.246	0.208	0.007

Alternatives	Seperation from best	Seperation from worst	Relative Closeness	Rank
RF	0.000	0.143	0.999	1
LR	0.143	0.000	0.000	3
XGB	0.052	0.092	0.639	2

Figure E.13 TOPSIS with Entropy Weights

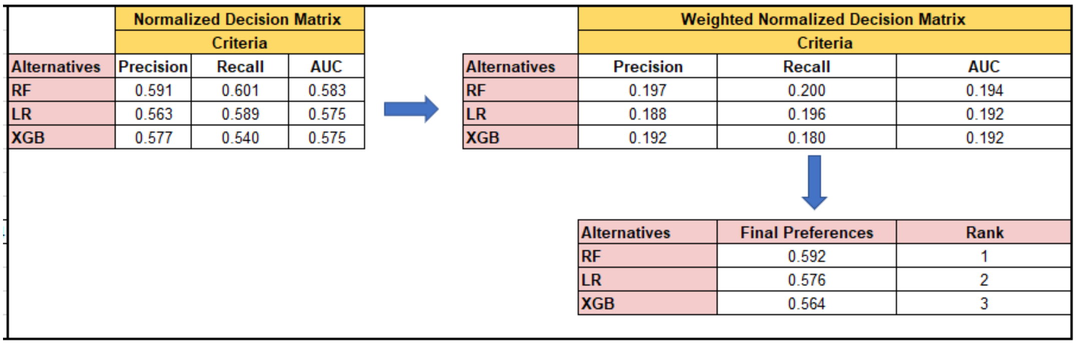
F. FOAM MOLDING CASE TOPSIS AND MULTI-MOORA RESULT

Multi-MOORA with Equal Weights:



	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.43	0.49	0.69
LR	0.41	0.48	0.68
XGB	0.42	0.44	0.68
Weights	0.33	0.33	0.33

Figure F.1 Decision Matrix

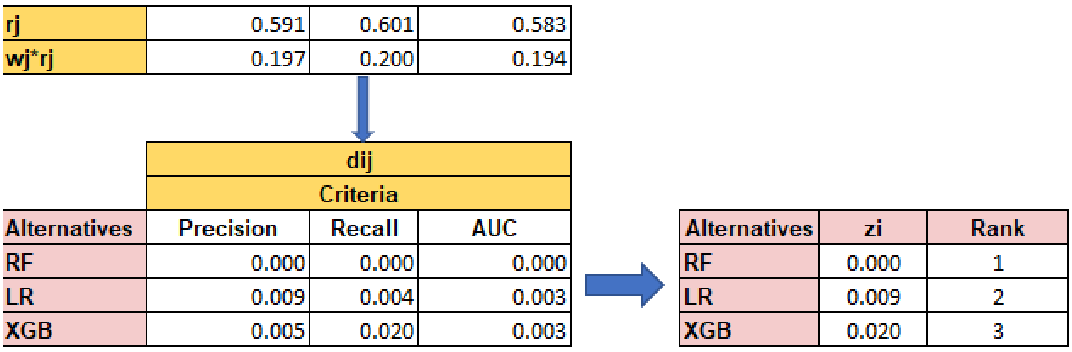


	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.591	0.601	0.583
LR	0.563	0.589	0.575
XGB	0.577	0.540	0.575

	Weighted Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.197	0.200	0.194
LR	0.188	0.196	0.192
XGB	0.192	0.180	0.192

Alternatives	Final Preferences	Rank
RF	0.592	1
LR	0.576	2
XGB	0.564	3

Figure F.2 Ratio System

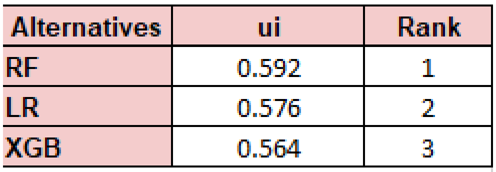


rj	0.591	0.601	0.583
wj*rj	0.197	0.200	0.194

	dij		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.000	0.000	0.000
LR	0.009	0.004	0.003
XGB	0.005	0.020	0.003

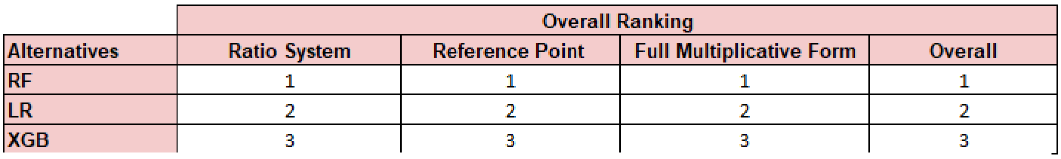
Alternatives	zi	Rank
RF	0.000	1
LR	0.009	2
XGB	0.020	3

Figure F.3 Reference Point Approach



Alternatives	ui	Rank
RF	0.592	1
LR	0.576	2
XGB	0.564	3

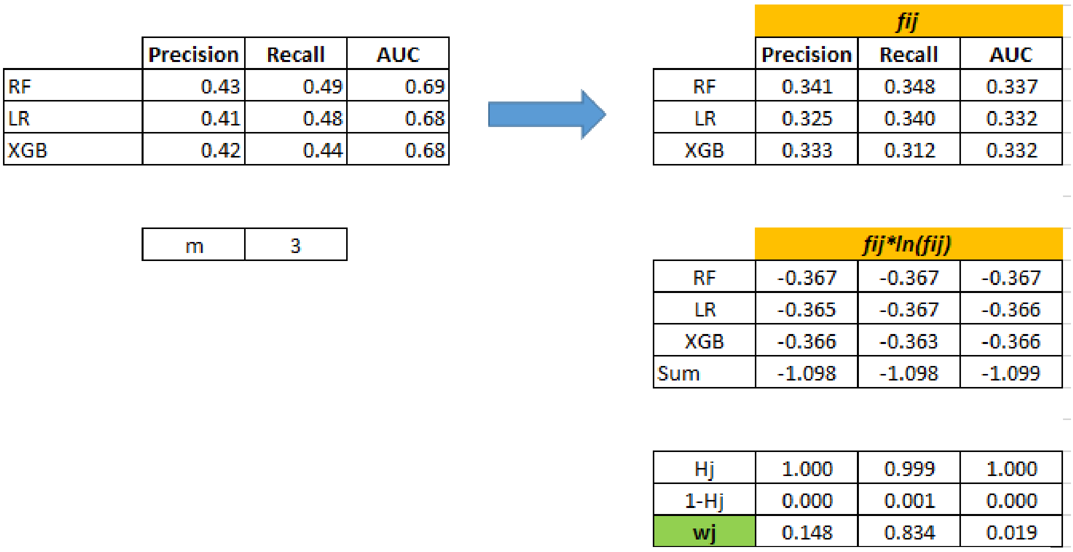
Figure F.4 Full Multiplicative Form



	Overall Ranking			
Alternatives	Ratio System	Reference Point	Full Multiplicative Form	Overall
RF	1	1	1	1
LR	2	2	2	2
XGB	3	3	3	3

Figure F.5 Overall Results of Multi-MOORA

Multi-MOORA with Entropy Weights:



	Precision	Recall	AUC
RF	0.43	0.49	0.69
LR	0.41	0.48	0.68
XGB	0.42	0.44	0.68

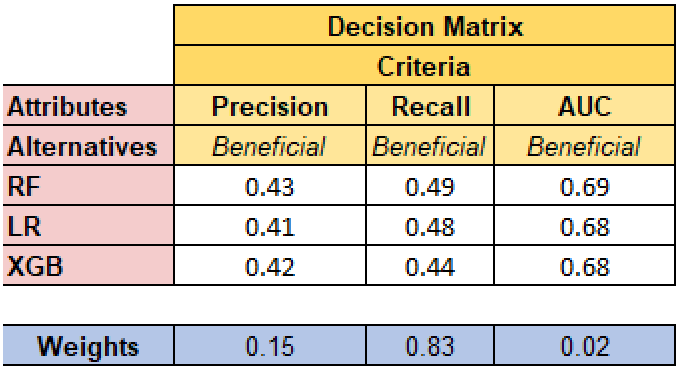
m	3

	fij		
	Precision	Recall	AUC
RF	0.341	0.348	0.337
LR	0.325	0.340	0.332
XGB	0.333	0.312	0.332

	*fij*ln(fij)*		
RF	-0.367	-0.367	-0.367
LR	-0.365	-0.367	-0.366
XGB	-0.366	-0.363	-0.366
Sum	-1.098	-1.098	-1.099

Hj	1.000	0.999	1.000
1-Hj	0.000	0.001	0.000
wj	0.148	0.834	0.019

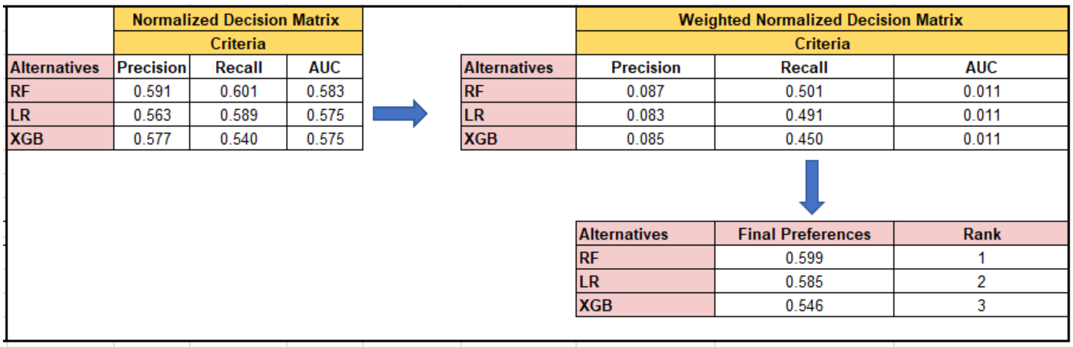
Figure F.6 Calculation of Entropy Weights



	Decision Matrix		
	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.43	0.49	0.69
LR	0.41	0.48	0.68
XGB	0.42	0.44	0.68

Weights	0.15	0.83	0.02

Figure F.7 Decision Matrix

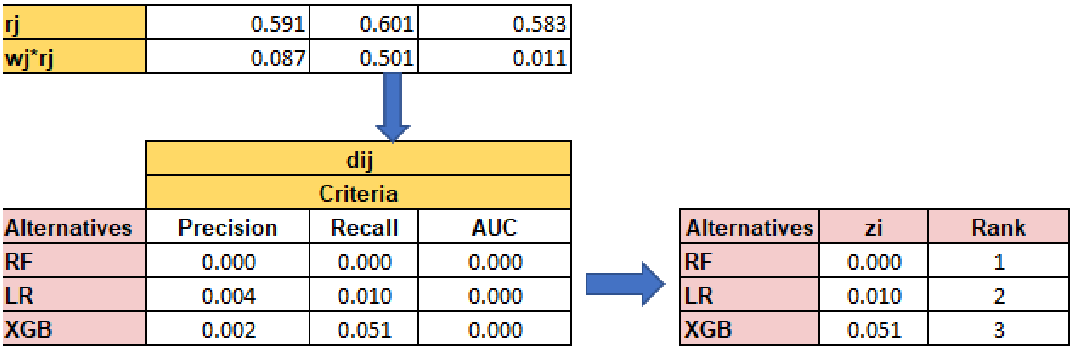


	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.591	0.601	0.583
LR	0.563	0.589	0.575
XGB	0.577	0.540	0.575

	Weighted Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.087	0.501	0.011
LR	0.083	0.491	0.011
XGB	0.085	0.450	0.011

Alternatives	Final Preferences	Rank
RF	0.599	1
LR	0.585	2
XGB	0.546	3

Figure F.8 Ratio System

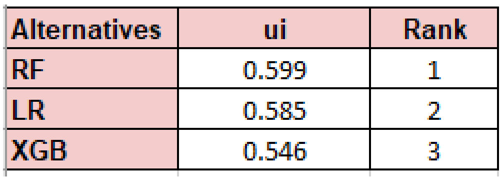


rj	0.591	0.601	0.583
wj*rj	0.087	0.501	0.011

	dij		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.000	0.000	0.000
LR	0.004	0.010	0.000
XGB	0.002	0.051	0.000

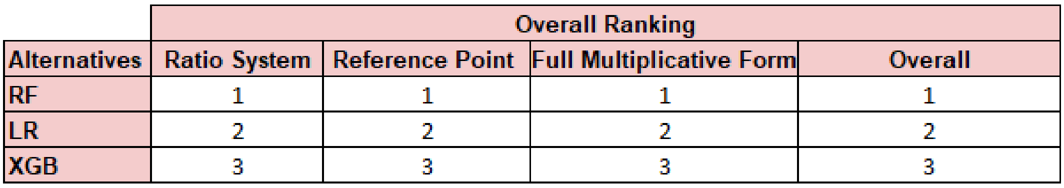
Alternatives	zi	Rank
RF	0.000	1
LR	0.010	2
XGB	0.051	3

Figure F.9 Reference Point Approach



Alternatives	ui	Rank
RF	0.599	1
LR	0.585	2
XGB	0.546	3

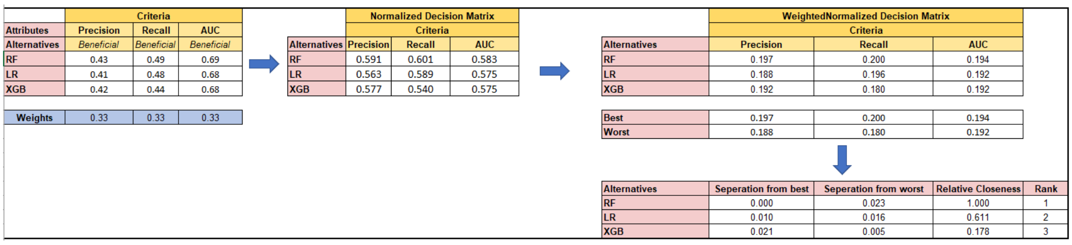
Figure F.10 Full Multiplicative Form



	Overall Ranking			
Alternatives	Ratio System	Reference Point	Full Multiplicative Form	Overall
RF	1	1	1	1
LR	2	2	2	2
XGB	3	3	3	3

Figure F.11 Overall Results of Multi-MOORA

TOPSIS with Equal Weights:



	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.43	0.49	0.69
LR	0.41	0.48	0.68
XGB	0.42	0.44	0.68
Weights	0.33	0.33	0.33

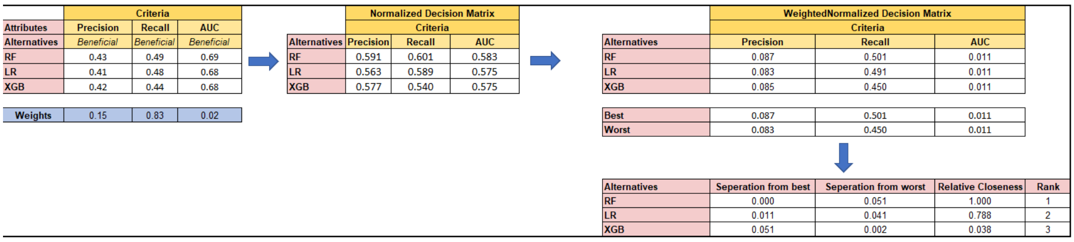
	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.591	0.601	0.583
LR	0.563	0.589	0.575
XGB	0.577	0.540	0.575

	WeightedNormalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.197	0.200	0.194
LR	0.188	0.196	0.192
XGB	0.192	0.180	0.192
Best	0.197	0.200	0.194
Worst	0.188	0.180	0.192

Alternatives	Seperation from best	Seperation from worst	Relative Closeness	Rank
RF	0.000	0.023	1.000	1
LR	0.010	0.016	0.611	2
XGB	0.021	0.005	0.178	3

Figure F.12 TOPSIS with Equal Weights

TOPSIS with Entropy Weights:



	Criteria		
Attributes	Precision	Recall	AUC
Alternatives	*Beneficial*	*Beneficial*	*Beneficial*
RF	0.43	0.49	0.69
LR	0.41	0.48	0.68
XGB	0.42	0.44	0.68
Weights	0.15	0.83	0.02

	Normalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.591	0.601	0.583
LR	0.563	0.589	0.575
XGB	0.577	0.540	0.575

	WeightedNormalized Decision Matrix		
	Criteria		
Alternatives	Precision	Recall	AUC
RF	0.087	0.501	0.011
LR	0.083	0.491	0.011
XGB	0.085	0.450	0.011
Best	0.087	0.501	0.011
Worst	0.083	0.450	0.011

Alternatives	Seperation from best	Seperation from worst	Relative Closeness	Rank
RF	0.000	0.051	1.000	1
LR	0.011	0.041	0.788	2
XGB	0.051	0.002	0.038	3

Figure F.13 TOPSIS with Entropy Weights

G. SENSITIVITY ANALYSIS RESULTS FOR TOPSIS

Table G.1 Sensitivity Analysis Results for Surface Defect Data

Weight			**Rank**
Precision	**Recall**	**AUC**	
0.033	0.483	0.483	XGB > RF > LR
0.133	0.433	0.433	XGB > RF > LR
0.233	0.383	0.383	XGB > RF > LR
0.333	0.333	0.333	XGB > RF > LR
0.433	0.283	0.283	XGB > RF > LR
0.533	0.233	0.233	XGB > RF > LR
0.633	0.183	0.183	XGB > RF > LR
0.733	0.133	0.133	XGB > RF > LR
0.833	0.083	0.083	XGB > RF > LR
0.933	0.033	0.033	XGB > RF > LR
0.483	0.033	0.483	XGB > RF > LR
0.433	0.133	0.433	XGB > RF > LR
0.383	0.233	0.383	XGB > RF > LR
0.283	0.433	0.283	XGB > RF > LR
0.233	0.533	0.233	XGB > RF > LR
0.183	0.633	0.183	XGB > RF > LR

Table G.1 (cont'd) Sensitivity Analysis Results for Surface Defect Data

0.133	0.733	0.133	XGB > RF > LR
0.083	0.833	0.083	XGB > RF > LR
0.033	0.933	0.033	XGB > RF > LR
0.483	0.483	0.033	XGB > RF > LR
0.433	0.433	0.133	XGB > RF > LR
0.383	0.383	0.233	XGB > RF > LR
0.283	0.283	0.433	XGB > RF > LR
0.233	0.233	0.533	XGB > RF > LR
0.183	0.183	0.633	XGB > RF > LR
0.133	0.133	0.733	XGB > RF > LR
0.083	0.083	0.833	XGB > RF > LR
0.033	0.033	0.933	XGB > RF > LR
0.940	0.000	0.060	XGB > RF > LR
0.840	0.000	0.160	XGB > RF > LR
0.740	0.000	0.260	XGB > RF > LR
0.640	0.000	0.360	XGB > RF > LR
0.540	0.000	0.460	XGB > RF > LR
0.440	0.000	0.560	XGB > RF > LR
0.340	0.000	0.660	XGB > RF > LR
0.240	0.000	0.760	XGB > RF > LR

Table G.1 (cont'd) Sensitivity Analysis Results for Surface Defect Data

0.140	0.000	0.860	XGB > RF > LR
0.040	0.000	0.960	XGB > RF > LR
0.666	0.100	0.234	XGB > RF > LR
0.592	0.200	0.208	XGB > RF > LR
0.518	0.300	0.182	XGB > RF > LR
0.444	0.400	0.156	XGB > RF > LR
0.370	0.500	0.130	XGB > RF > LR
0.296	0.600	0.104	XGB > RF > LR
0.222	0.700	0.078	XGB > RF > LR
0.148	0.800	0.052	XGB > RF > LR
0.074	0.900	0.026	XGB > RF > LR
0.940	0.000	0.060	XGB > RF > LR
0.840	0.000	0.160	XGB > RF > LR
0.740	0.000	0.260	XGB > RF > LR
0.640	0.000	0.360	XGB > RF > LR
0.540	0.000	0.460	XGB > RF > LR
0.440	0.000	0.560	XGB > RF > LR
0.340	0.000	0.660	XGB > RF > LR
0.240	0.000	0.760	XGB > RF > LR
0.040	0.000	0.960	XGB > RF > LR

Table G.2 Sensitivity Analysis Results for Duplicator Data

Weight			**Rank**
Precision	**Recall**	**AUC**	
0.033	0.483	0.483	XGB > RF > LR
0.133	0.433	0.433	XGB > RF > LR
0.233	0.383	0.383	XGB > RF > LR
0.333	0.333	0.333	XGB > RF > LR
0.433	0.283	0.283	XGB > RF > LR
0.533	0.233	0.233	XGB > RF > LR
0.633	0.183	0.183	XGB > RF > LR
0.733	0.133	0.133	XGB > RF > LR
0.833	0.083	0.083	XGB > RF > LR
0.933	0.033	0.033	XGB > RF > LR
0.483	0.033	0.483	XGB > RF > LR
0.433	0.133	0.433	XGB > RF > LR
0.383	0.233	0.383	XGB > RF > LR
0.283	0.433	0.283	XGB > RF > LR
0.233	0.533	0.233	XGB > RF > LR
0.183	0.633	0.183	XGB > RF > LR
0.133	0.733	0.133	XGB > RF > LR

Table G.2 (cont'd) Sensitivity Analysis Results for Duplicator Data

0.083	0.833	0.083	XGB > RF > LR
0.033	0.933	0.033	XGB > RF > LR
0.483	0.483	0.033	XGB > RF > LR
0.433	0.433	0.133	XGB > RF > LR
0.383	0.383	0.233	XGB > RF > LR
0.283	0.283	0.433	XGB > RF > LR
0.233	0.233	0.533	XGB > RF > LR
0.183	0.183	0.633	**RF > XGB > LR**
0.133	0.133	0.733	**RF > XGB > LR**
0.083	0.083	0.833	**RF > XGB > LR**
0.033	0.033	0.933	**RF > XGB > LR**
0.071	0.898	0.031	XGB > RF > LR
0.171	0.802	0.028	XGB > RF > LR
0.271	0.705	0.024	XGB > RF > LR
0.371	0.608	0.021	XGB > RF > LR
0.471	0.512	0.018	XGB > RF > LR
0.571	0.415	0.014	XGB > RF > LR
0.671	0.318	0.011	XGB > RF > LR
0.771	0.222	0.008	XGB > RF > LR

Table G.2 (cont'd) Sensitivity Analysis Results for Duplicator Data

0.871	0.125	0.004	XGB > RF > LR
0.971	0.028	0.001	XGB > RF > LR
0.001	0.998	0.001	XGB > RF > LR
0.140	0.798	0.062	XGB > RF > LR
0.210	0.698	0.092	XGB > RF > LR
0.279	0.598	0.123	XGB > RF > LR
0.348	0.498	0.153	XGB > RF > LR
0.418	0.398	0.184	XGB > RF > LR
0.487	0.298	0.214	XGB > RF > LR
0.557	0.198	0.245	XGB > RF > LR
0.626	0.098	0.276	XGB > RF > LR
0.063	0.806	0.131	XGB > RF > LR
0.056	0.713	0.231	XGB > RF > LR
0.049	0.620	0.331	XGB > RF > LR
0.041	0.527	0.431	XGB > RF > LR
0.034	0.435	0.531	**RF > XGB > LR**
0.027	0.342	0.631	**RF > XGB > LR**
0.020	0.249	0.731	**RF > XGB > LR**
0.012	0.157	0.831	**RF > XGB > LR**
0.005	0.064	0.931	**RF > XGB > LR**

Table G.3 Sensitivity Analysis Results for Inkjet Printer Data

Weight			Rank
Precision	**Recall**	**AUC**	
0.033	0.483	0.483	RF > XGB > LR
0.133	0.433	0.433	RF > XGB > LR
0.233	0.383	0.383	RF > XGB > LR
0.333	0.333	0.333	RF > XGB > LR
0.433	0.283	0.283	RF > XGB > LR
0.533	0.233	0.233	RF > XGB > LR
0.633	0.183	0.183	RF > XGB > LR
0.733	0.133	0.133	RF > XGB > LR
0.833	0.083	0.083	RF > XGB > LR
0.933	0.033	0.033	RF > XGB > LR
0.483	0.033	0.483	RF > XGB > LR
0.433	0.133	0.433	RF > XGB > LR
0.383	0.233	0.383	RF > XGB > LR
0.283	0.433	0.283	RF > XGB > LR
0.233	0.533	0.233	RF > XGB > LR
0.183	0.633	0.183	RF > XGB > LR
0.133	0.733	0.133	RF > XGB > LR
0.083	0.833	0.083	RF > XGB > LR

Table G.3 (cont'd) Sensitivity Analysis Results for Inkjet Printer Data

0.033	0.933	0.033	RF > XGB > LR
0.483	0.483	0.033	RF > XGB > LR
0.433	0.433	0.133	RF > XGB > LR
0.383	0.383	0.233	RF > XGB > LR
0.283	0.283	0.433	RF > XGB > LR
0.233	0.233	0.533	RF > XGB > LR
0.183	0.183	0.633	RF > XGB > LR
0.133	0.133	0.733	RF > XGB > LR
0.083	0.083	0.833	RF > XGB > LR
0.033	0.033	0.933	**XGB > RF > LR**
0.038	0.935	0.027	RF > XGB > LR
0.138	0.838	0.024	RF > XGB > LR
0.238	0.741	0.022	RF > XGB > LR
0.338	0.643	0.019	RF > XGB > LR
0.438	0.546	0.016	RF > XGB > LR
0.538	0.449	0.013	RF > XGB > LR
0.638	0.352	0.010	RF > XGB > LR
0.738	0.255	0.007	RF > XGB > LR
0.838	0.157	0.005	RF > XGB > LR
0.938	0.060	0.002	RF > XGB > LR

Table G.3 (cont'd) Sensitivity Analysis Results for Inkjet Printer Data

0.929	0.049	0.023	RF > XGB > LR
0.831	0.149	0.020	RF > XGB > LR
0.733	0.249	0.018	RF > XGB > LR
0.636	0.349	0.015	RF > XGB > LR
0.538	0.449	0.013	RF > XGB > LR
0.440	0.549	0.011	RF > XGB > LR
0.343	0.649	0.008	RF > XGB > LR
0.245	0.749	0.006	RF > XGB > LR
0.147	0.849	0.004	RF > XGB > LR
0.050	0.949	0.001	RF > XGB > LR
0.483	0.403	0.113	RF > XGB > LR
0.429	0.358	0.213	RF > XGB > LR
0.374	0.313	0.313	RF > XGB > LR
0.320	0.267	0.413	RF > XGB > LR
0.265	0.222	0.513	RF > XGB > LR
0.211	0.176	0.613	RF > XGB > LR
0.156	0.131	0.713	RF > XGB > LR
0.102	0.085	0.813	RF > XGB > LR
0.047	0.040	0.913	**XGB > RF > LR**

Table G.4 Sensitivity Analysis Results for Foam Molding Data

Weight			**Rank**
Precision	**Recall**	**AUC**	
0.033	0.483	0.483	RF > LR > XGB
0.133	0.433	0.433	RF > LR > XGB
0.233	0.383	0.383	RF > LR > XGB
0.333	0.333	0.333	RF > LR > XGB
0.433	0.283	0.283	RF > LR > XGB
0.533	0.233	0.233	**RF > XGB > LR**
0.633	0.183	0.183	**RF > XGB > LR**
0.733	0.133	0.133	**RF > XGB > LR**
0.833	0.083	0.083	**RF > XGB > LR**
0.933	0.033	0.033	**RF > XGB > LR**
0.483	0.033	0.483	**RF > XGB > LR**
0.433	0.133	0.433	**RF > XGB > LR**
0.383	0.233	0.383	RF > LR > XGB
0.283	0.433	0.283	RF > LR > XGB
0.233	0.533	0.233	RF > LR > XGB
0.183	0.633	0.183	RF > LR > XGB
0.133	0.733	0.133	RF > LR > XGB
0.083	0.833	0.083	RF > LR > XGB

Table G.4 (cont'd) Sensitivity Analysis Results for Foam Molding Data

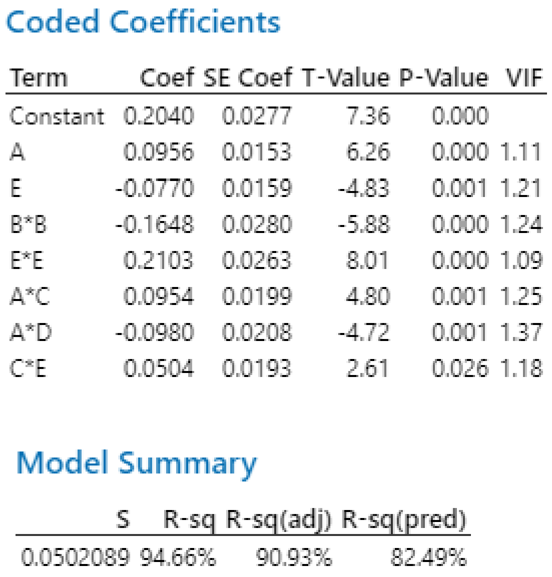
0.033	0.933	0.033	RF > LR > XGB
0.483	0.483	0.033	RF > LR > XGB
0.433	0.433	0.133	RF > LR > XGB
0.383	0.383	0.233	RF > LR > XGB
0.283	0.283	0.433	RF > LR > XGB
0.233	0.233	0.533	RF > LR > XGB
0.183	0.183	0.633	RF > LR > XGB
0.133	0.133	0.733	RF > LR > XGB
0.083	0.083	0.833	RF > LR > XGB
0.033	0.033	0.933	RF > LR > XGB
0.048	0.932	0.021	RF > LR > XGB
0.148	0.834	0.019	RF > LR > XGB
0.248	0.736	0.016	RF > LR > XGB
0.348	0.638	0.014	RF > LR > XGB
0.448	0.540	0.012	RF > LR > XGB
0.548	0.442	0.010	RF > LR > XGB
0.648	0.345	0.008	RF > LR > XGB
0.748	0.247	0.005	RF > LR > XGB
0.848	0.149	0.003	**RF > XGB > LR**
0.948	0.051	0.001	**RF > XGB > LR**

Table G.4 (cont'd) Sensitivity Analysis Results for Foam Molding Data

0.858	0.034	0.108	**RF > XGB > LR**
0.770	0.134	0.097	**RF > XGB > LR**
0.681	0.234	0.086	RF > LR > XGB
0.592	0.334	0.074	RF > LR > XGB
0.503	0.434	0.063	RF > LR > XGB
0.414	0.534	0.052	RF > LR > XGB
0.325	0.634	0.041	RF > LR > XGB
0.237	0.734	0.030	RF > LR > XGB
0.148	0.834	0.019	RF > LR > XGB
0.059	0.934	0.007	RF > LR > XGB
0.133	0.749	0.119	RF > LR > XGB
0.118	0.664	0.219	RF > LR > XGB
0.103	0.579	0.319	RF > LR > XGB
0.088	0.494	0.419	RF > LR > XGB
0.072	0.409	0.519	RF > LR > XGB
0.057	0.324	0.619	RF > LR > XGB
0.042	0.239	0.719	RF > LR > XGB
0.027	0.154	0.819	RF > LR > XGB
0.012	0.069	0.919	RF > LR > XGB

H. SURFACE DEFECT CASE REGRESSION MODELS RESULTS

Linear Regression Results for COV



Coded Coefficients

Term	Coef	SE Coef	T-Value	P-Value	VIF
Constant	0.2040	0.0277	7.36	0.000	
A	0.0956	0.0153	6.26	0.000	1.11
E	-0.0770	0.0159	-4.83	0.001	1.21
B*B	-0.1648	0.0280	-5.88	0.000	1.24
E*E	0.2103	0.0263	8.01	0.000	1.09
A*C	0.0954	0.0199	4.80	0.001	1.25
A*D	-0.0980	0.0208	-4.72	0.001	1.37
C*E	0.0504	0.0193	2.61	0.026	1.18

Model Summary

S	R-sq	R-sq(adj)	R-sq(pred)
0.0502089	94.66%	90.93%	82.49%

Figure H.1 Model Summary for COV Model

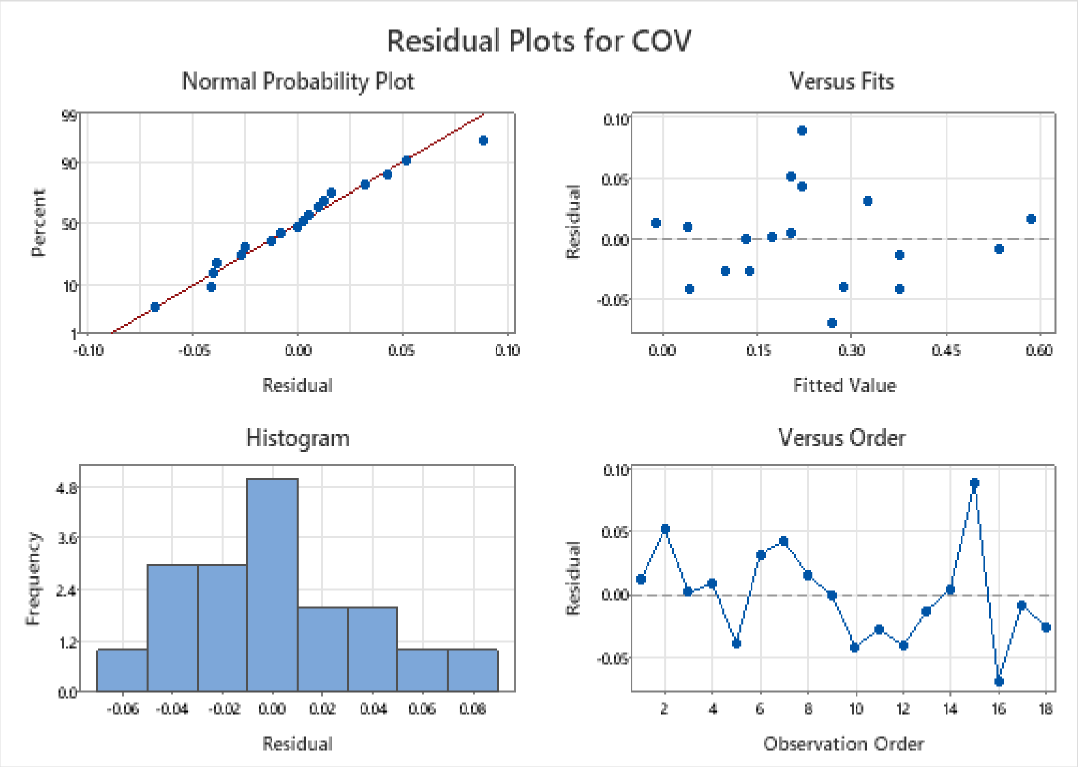


Figure H.2 Residual Plots for COV Model

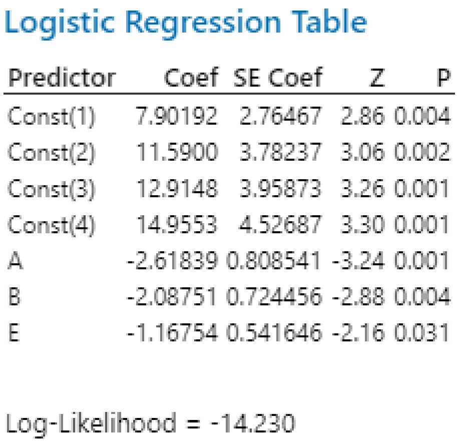
Since all p-values are less than 0.05, all of the factors are significant according to Figure H.1 Variation inflation factor (VIF) values are changing from 1 to 1.34. This shows that there is no multicollinearity (correlation between factors) problem. Minitab 2020, define VIF as if it is 1 there is no correlation, and if it is less than 5, there is moderate correlation between predictors. All VIF values are close to 1. The SE Coef column shows the standard error of the coefficient, the smaller the error, the more accurate the prediction. All of the standard errors are very close 0.

S value in Figure H.1 is used to understand how far the predicted values fall from the real values. The lower the S, the better the model explains COV (Minitab 2020). Our S value is 0.0502089 which very close to 0, so this model adequately explains the COV.

94.66% of variation in the COV is explained by this model according to R^2. Adding more independent variables to the model increases R^2, for this reason adjusted- R^2

is also considered which equals to 90.93%. The model is evaluated based on the prediction ability with the predicted- R^2 which is also at satisfying level. In Figure H.2, residuals generally follow a straight line, it can be said that residuals are normally distributed. Also, no pattern is recognized in residuals versus fit plot, and data points are randomly distributed both sides of 0, so residuals have constant variance. In residuals versus order plots no pattern is observed, so residuals are independent from each other.

Ordinal Logistic Regression Results for Median

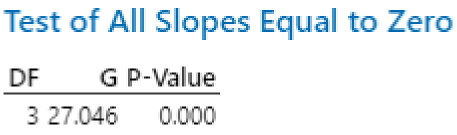


Logistic Regression Table

Predictor	Coef	SE Coef	Z	P
Const(1)	7.90192	2.76467	2.86	0.004
Const(2)	11.5900	3.78237	3.06	0.002
Const(3)	12.9148	3.95873	3.26	0.001
Const(4)	14.9553	4.52687	3.30	0.001
A	-2.61839	0.808541	-3.24	0.001
B	-2.08751	0.724456	-2.88	0.004
E	-1.16754	0.541646	-2.16	0.031

Log-Likelihood = -14.230

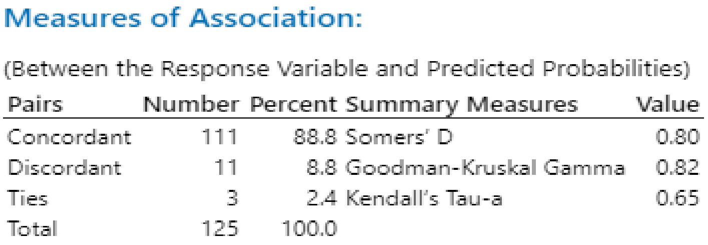
Figure H.3 Logistic Regression Table for Median Model



Test of All Slopes Equal to Zero

DF	G	P-Value
3	27.046	0.000

Figure H.4 Test of All Slopes for Median Model



Measures of Association:

(Between the Response Variable and Predicted Probabilities)

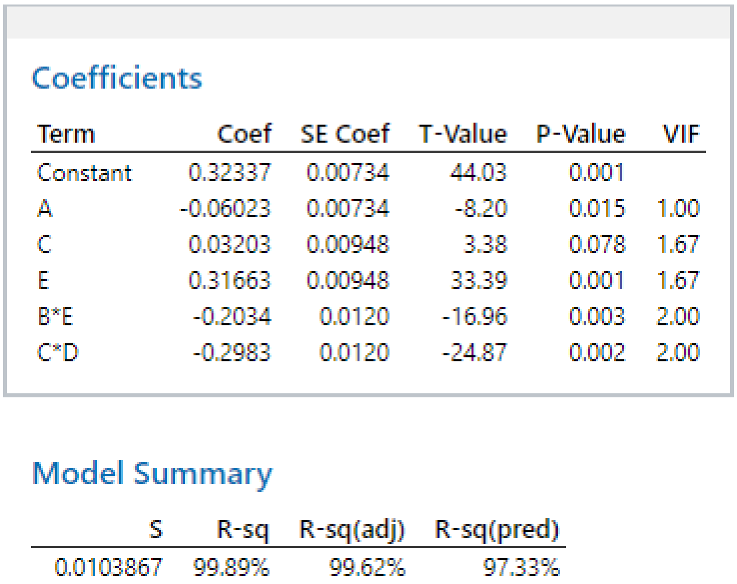
Pairs	Number	Percent	Summary Measures	Value
Concordant	111	88.8	Somers' D	0.80
Discordant	11	8.8	Goodman-Kruskal Gamma	0.82
Ties	3	2.4	Kendall's Tau-a	0.65
Total	125	100.0		

Figure H.5 Measures of Association for Median Model

In Figure H.3 all p-values are less than 0.05 that means all predictors are significant. Test of all slopes shows that there is at least one predictor that has a statistically significant relationship with response (median), p-value is 0. That is, the null hypothesis that all coefficients are nearly equal to zero can be rejected. Figure H.4 shows the measure of association. Also, Somers' D, Goodman-Kruskal Gamma, and Kendall's Tau-a are useful when comparing predictive performance of models. Higher values mean better performance. It can be said that the Ordinal Logistic Regression model has good predictive performance.

I. INKJET PRINTER CASE REGRESSION MODELS RESULTS

Linear Regression Results for COV



Coefficients

Term	Coef	SE Coef	T-Value	P-Value	VIF
Constant	0.32337	0.00734	44.03	0.001	
A	-0.06023	0.00734	-8.20	0.015	1.00
C	0.03203	0.00948	3.38	0.078	1.67
E	0.31663	0.00948	33.39	0.001	1.67
B*E	-0.2034	0.0120	-16.96	0.003	2.00
C*D	-0.2983	0.0120	-24.87	0.002	2.00

Model Summary

S	R-sq	R-sq(adj)	R-sq(pred)
0.0103867	99.89%	99.62%	97.33%

Figure I.1 Model Summary for COV Model

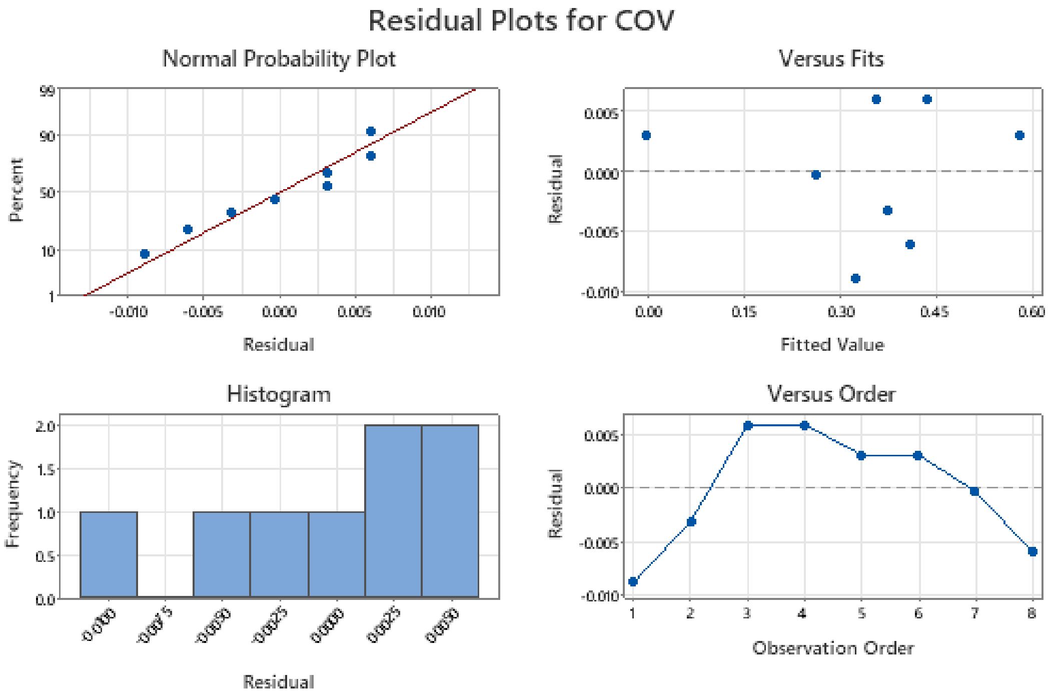
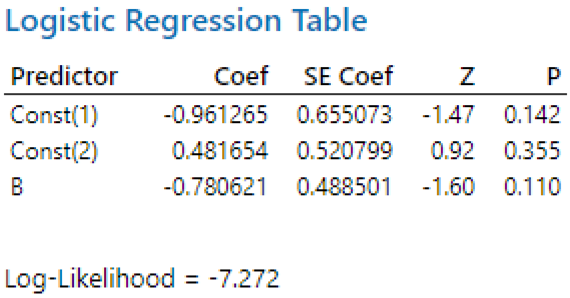


Figure I.2 Residual Plots for COV Model

Since all p-values are less than 0.05, all of the factors are significant according to Figure I.1. VIF in the model changes between 1 and 2. All of the standard errors are very close 0. Also, S value is 0.0103867 which very close to 0, this model adequately explains the COV.

99.89% of variation in the COV is explained by this model according to R^2. Moreover adjusted- R^2 and predicted- R^2 are also at satisfying levels. In Figure I.2, residuals generally follow a straight line indicating approximately that normality assumption is satisfied. Also, no pattern is recognized in residuals versus fit plot, so residuals are randomly distributed and have constant variance. There is an increasing and then decreasing pattern in residual versus order plot but only 8 observations are used. Moreover, Minitab (2020), mentioned using histogram plot of residuals is better when there are 20 or more data points in the dataset.

Ordinal Logistic Regression Results for Median

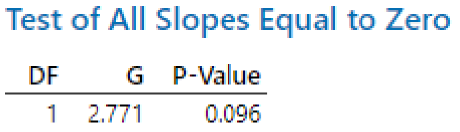


Logistic Regression Table

Predictor	Coef	SE Coef	Z	P
Const(1)	-0.961265	0.655073	-1.47	0.142
Const(2)	0.481654	0.520799	0.92	0.355
B	-0.780621	0.488501	-1.60	0.110

Log-Likelihood = -7.272

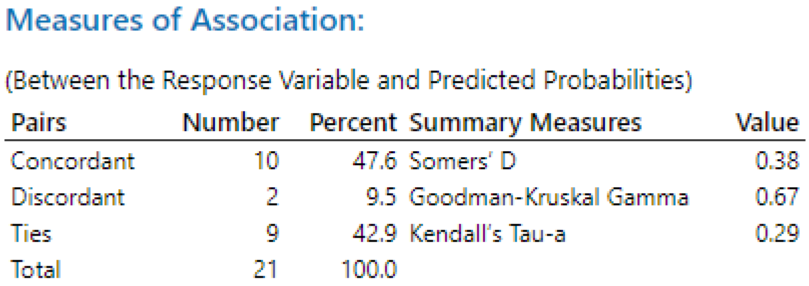
Figure I.3 Logistic Regression Table



Test of All Slopes Equal to Zero

DF	G	P-Value
1	2.771	0.096

Figure I.4 Test of All Slopes for Median Model



Measures of Association:

(Between the Response Variable and Predicted Probabilities)

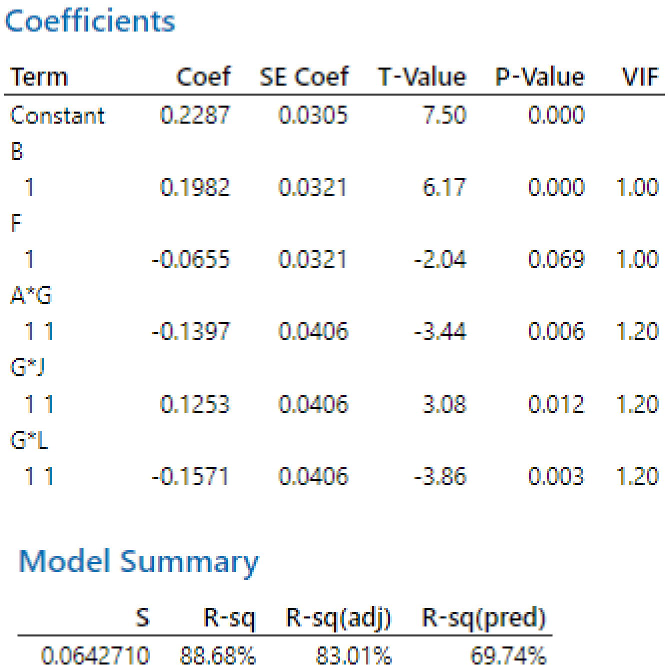
Pairs	Number	Percent	Summary Measures	Value
Concordant	10	47.6	Somers' D	0.38
Discordant	2	9.5	Goodman-Kruskal Gamma	0.67
Ties	9	42.9	Kendall's Tau-a	0.29
Total	21	100.0		

Figure I.5 Measures of Association for Median Model

In Figure I.3, p-value for factor B is close to 0.1, adding more terms and interactions does not improve the performance. Therefore, the Ordinal Logistic Regression model is fitted using only this term. Test of all slopes shows that there is at least one predictor that has a statistically significant relationship with response (median), p-value is less than 0.1. Result of Pearson and Deviance Goodness-of-Fit test results not included because Minitab (2020) mentioned that these tests are not useful when the number of unique values is nearly equal to the number of observations. In this thesis, for three of the case study, all the data points for the median model are unique and equal to the number of observations. Figure I.4 shows the measure of association. Concordant pairs are 10 out of 21 pairs that prediction performance of the model is good. Also, Somers' D shows the percentage difference between concordant and discordant, and ties are included, while Goodman-Kruskal Gamma measure same but do not include ties. There are 9 ties, therefore the result of Goodman-Kruskal Gamma is higher than Somers' D. Kendall's Tau-a is the measure difference between concordant and discordant pairs out of all possible pairs (Minitab, 2020). It can be said that this Ordinal Logistic Regression model has moderate performance when it compared with the Ordinal Logistic Regression model in the surface defect case.

J. DUPLICATOR CASE REGRESSION MODELS RESULTS

Linear Regression Results for COV



Coefficients

Term	Coef	SE Coef	T-Value	P-Value	VIF
Constant	0.2287	0.0305	7.50	0.000	
B					
1	0.1982	0.0321	6.17	0.000	1.00
F					
1	-0.0655	0.0321	-2.04	0.069	1.00
A*G					
1 1	-0.1397	0.0406	-3.44	0.006	1.20
G*J					
1 1	0.1253	0.0406	3.08	0.012	1.20
G*L					
1 1	-0.1571	0.0406	-3.86	0.003	1.20

Model Summary

S	R-sq	R-sq(adj)	R-sq(pred)
0.0642710	88.68%	83.01%	69.74%

Figure J.1 Model Summary for COV Model

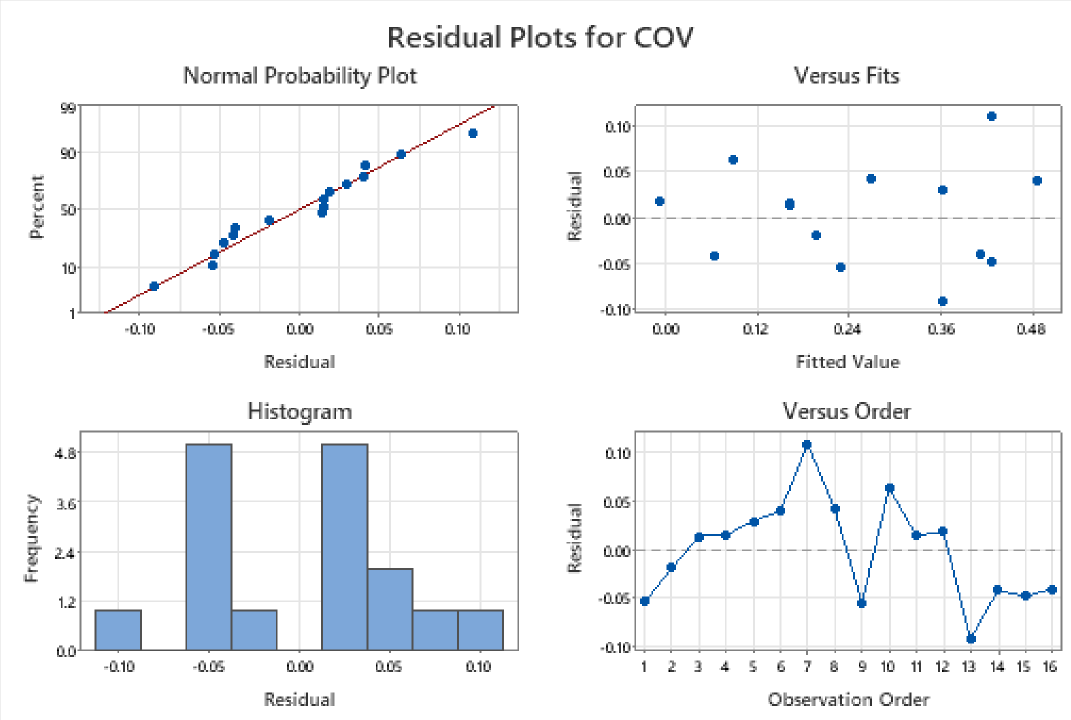
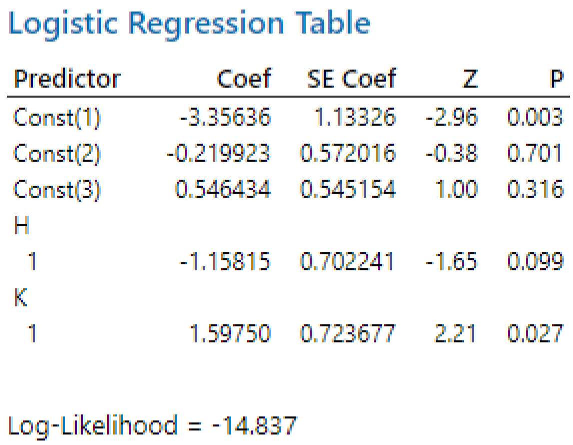


Figure J.2 Residual Plots for COV Model

All p-values are less than 0.05, it means all of the factors are significant according to Figure J.1. VIF values in the model changes between 1 and 1.2, therefore multicollinearity problem is avoided. All of the standard errors are very close 0. Also, S value is 0.0642710 which very close to 0, this model adequately explains the COV.

88.68% of variation in the COV is explained by this model according to R^2. Moreover adjusted- R^2 is parallel with R^2 and predicted- R^2 are moderate levels, but any other terms does not improve predicted-R^2. Figure J.2 shows that residuals generally follow a straight line, indicating that normality assumption is satisfied. Also, no pattern is recognized in residuals versus fit plot, so residuals are randomly distributed and have constant variance, also residuals are independent from each other according to residuals versus order plot. Evaluating histogram of residual is not effective because there are 16 observations in the dataset.

<u>Ordinal Logistic Regression Results for Median</u>

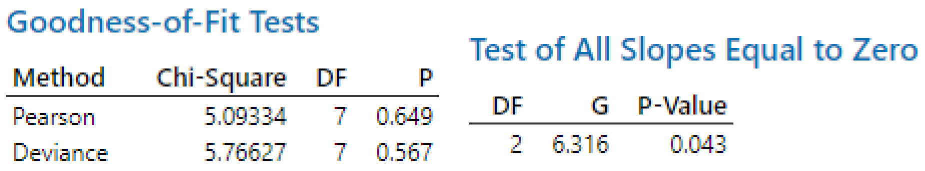


Logistic Regression Table

Predictor	Coef	SE Coef	Z	P
Const(1)	-3.35636	1.13326	-2.96	0.003
Const(2)	-0.219923	0.572016	-0.38	0.701
Const(3)	0.546434	0.545154	1.00	0.316
H				
1	-1.15815	0.702241	-1.65	0.099
K				
1	1.59750	0.723677	2.21	0.027

Log-Likelihood = -14.837

Figure J.3 Logistic Regression Table



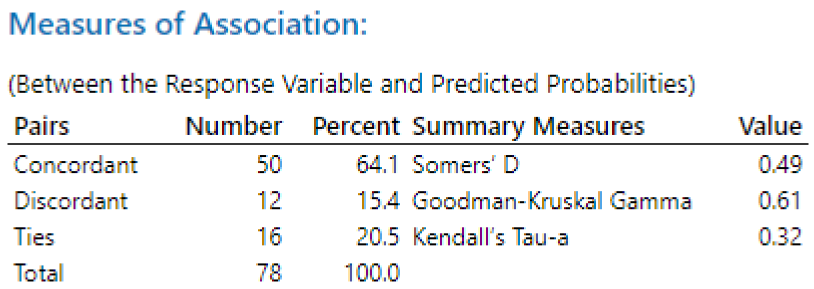
Goodness-of-Fit Tests

Method	Chi-Square	DF	P
Pearson	5.09334	7	0.649
Deviance	5.76627	7	0.567

Test of All Slopes Equal to Zero

DF	G	P-Value
2	6.316	0.043

Figure J.4 Test of All Slopes for Median Model



Measures of Association:

(Between the Response Variable and Predicted Probabilities)

Pairs	Number	Percent	Summary Measures	Value
Concordant	50	64.1	Somers' D	0.49
Discordant	12	15.4	Goodman-Kruskal Gamma	0.61
Ties	16	20.5	Kendall's Tau-a	0.32
Total	78	100.0		

Figure J.5 Measures of Association for Median Model

In Figure J.3, *p*-values for factor H and K are less than 0.1, so they are statistically significant. Test of all slopes shows that there is at least one predictor that has a statistically significant relationship with response (median), *p*-value is less than 0.1. *p*-values for Figure J.5. shows the measure of association. Concordant pairs are 50 out of 78 pairs that predict the performance of the model is good. There are 16 ties, so the result of Goodman-Kruskal Gamma is higher than Somers’ D. It can be said that this Ordinal Logistic Regression model has good performance.