ROBUST PARAMETER DESIGN OF PRODUCTS AND PROCESSES WITH AN ORDINAL CATEGORICAL RESPONSE USING RANDOM FORESTS

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

BY

SEÇİL GÜLBUDAK DİL

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

JUNE 2018

Approval of the thesis:

ROBUST PARAMETER DESIGN OF PRODUCTS AND PROCESSES WITH AN ORDINAL CATEGORICAL RESPONSE USING RANDOM FORESTS

submitted by **SEÇİL GÜLBUDAK DİL** in partial fulfillment of the requirement for the degree of **Master of Science in Industrial Engineering Department, Middle East Technical University** by,

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

Prof. Dr. Yasemin Serin
Head of Department, **Industrial Engineering** ____________________

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

Examining Committee Members

Prof. Dr. Esra Karasakal
Industrial Engineering Dept., METU ____________________

Prof. Dr. Gülser Köksal
Industrial Engineering Dept., METU ____________________

Assoc. Prof. Dr. Sinan Gürel
Industrial Engineering Dept., METU ____________________

Assoc. Prof. Dr. Cem İyigün
Industrial Engineering Dept., METU ____________________

Assist. Prof. Dr. Leman Esra Dolgun
Industrial Engineering Dept., Eskişehir Technical University ____________________

Date: ____________________

ABSTRACT

ROBUST PARAMETER DESIGN OF PRODUCTS AND PROCESSES WITH AN ORDINAL CATEGORICAL RESPONSE USING RANDOM FORESTS

Gülbudak Dil, Seçil
M.S., Department of Industrial Engineering
Supervisor: Prof. Dr. Gülser Köksal

June 2018, 201 pages

In industrial organizations, manufacturers aim to achieve target product performance with minimum variation. For that reason, finding optimal settings of product and process design parameters that make it possible to consistently achieve target product performance is an important design problem. In this study, we propose an alternative method to solve this problem for the case of an ordinal categorical product/process response. The method utilizes Random Forest (RF) for modelling mean and variance of the response at a given set of parameter settings. The method uses different weighting strategies of Random Forest, and it is applied on three case problems. Two of the case problems are of the larger-the-better type, and the other one is of the smaller-the-better type. In addition, obtained results are compared with those of previous studies that used the same data sets. In comparing the results, classification performance, probability of observing target class, and both location and dispersion of results are considered. Advantages and disadvantages of the proposed method are discussed.

Key words: Robust Parameter Design, Design Parameter Optimization, Random Forest, Ordinal Categorical Response

ÖZ

SIRALI KATEGORİK YANIT VEREN ÜRÜNLERİN VE SÜREÇLERİN RASSAL ORMANLARI KULLANARAK GÜRBÜZ PARAMETRE TASARIMI

Gülbudak Dil, Seçil
Yüksek Lisans, Endüstri Mühendisliği
Tez Yöneticisi: Prof. Dr. Gülser Köksal

Haziran 2018, 201 sayfa

Endüstriyel organizasyonlarda, üreticiler ürün performansını hedefte elde ederken değişkenliğin de en düşük seviyede olmasını isterler. Bu nedenle, ürün ve süreç tasarım parametrelerinin, kararlı bir şekilde her seferinde hedef ürün performansını verecek en iyi değerlerini elde etmek önemli bir tasarım problemidir. Bu çalışmada, sıralı kategorik yanıtı olan bir ürün veya süreç söz konusu olduğunda bu problemi çözmek için alternatif bir yöntem önerilmiştir. Yöntem, belirli parametre değerleri için yanıtın ortalamasını ve varyansını modellemek amacıyla Rassal Ormanları kullanmaktadır. Yöntem, Rassal Ormanların farklı ağırlıklandırma stratejilerini kullanmış ve üç vaka problemine uygulanmıştır. Bu vakalardan ikisi en büyük-en-iyi; diğeri ise en küçük-en-iyi tipindedir. Ayrıca, elde edilen sonuçlar, aynı verileri kullanan daha önceki çalışmaların sonuçlarıyla karşılaştırılmıştır. Sonuçların karşılaştırmasında, sınıflandırma performansı, hedeflenen kategorinin gözlemlenme olasılığı ile konum ve dağılım sonuçları dikkate alınmıştır. Önerilen yöntemin yararları ve zayıf olabilecek yanları tartışılmıştır.

Anahtar kelimeler: Gürbüz Parametre Tasarımı, Tasarım Parametre Optimizasyonu, Rassal Ormanlar, Sıralı Kategorik Yanıt

To my family and my beloved husband

ACKNOWLEDGEMENTS

First of all, I would like to express my special appreciation and thanks to my supervisor Prof. Dr. Gülser Köksal for her guidance, advice, criticism, encouragements, and insight throughout the research. Your advice on both research as well as on my career have been priceless.

I would also thank Güncel Vara and Aslı Özahi for their help, interest, explanations, and patiently answering all of my questions. I would especially like to express my deepest gratitude to Emine Gündoğdu for her endless and invaluable helps.

I would also like to express my deepest thanks to my mother Gülüzar Gülbudak, my father Salman Gülbudak, and my brothers Seçkin Gülbudak and Berkan Gülbudak for all of the sacrifices that they have made on my behalf, their invaluable support, endless understanding and especially patience.

Finally, my special thanks to my beloved husband Ayhan Dil for his love, motivation, help, support, encouragement, and patience. Words cannot express how grateful I am to him. Without his, it would have been impossible to complete this work. I would like to express my sincere gratitude to him making.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AA	Accumulation Analysis
Acc	Accuracy
AUC	Area Under the Receiver Operating Characteristic (ROC) Curve
EV	Expected Value
LR	Logistic Regression
LRMO	Logistic Regression Model Optimization
MisErr	Misclassification Error
OOB	Out of Bag
RF(s)	Random Forest(s)
RF_E	Random Forest with equally distributed weighting strategy
RF_L	Random Forest with linearly distributed weighting strategy
RF_P	Random Forest with piecewise linearly distributed weighting strategy
RF_X	Random Forest with exponentially distributed weighting strategy
RPD	Robust Parameter Design
SNR	Signal to Noise Ratio
SS	Scoring Scheme
Var	Variance
WPSS	Weighted Probability Scoring Scheme
WSNR	Weighted Signal-to-noise Ratio

CHAPTER 1

INTRODUCTION

In today's competitive markets, it is crucial for companies to continuously improve the design quality of products and processes. Variation in performance of products or processes is the main challenge to deal with in product/process design. Variability in quality characteristics of products/processes is often due to factors that we cannot control. Robust Parameter Design (RPD) approaches have been developed for more than forty years to minimize such variations through design. With these approaches, parameters of controllable design factors are adjusted so that the product (or process) becomes as insensitive as possible to the uncontrollable factors.

A quality characteristic (or response) of a product or process can be either quantitative or qualitative. RPD studies are largely focused on quantitative or continuous responses. A limited number of studies have been proposed for RPD involving qualitative responses, with each of these having their own deficiencies. Hence, there is still a need for a superior method to handle qualitative responses.

In this study, a new RPD approach is proposed for the case of a single categorical response. The method is based on the Random Forest method. This method can be used to predict the value of a response of a product or process when the response is continuous or categorical. It is not important for this method whether design parameters or factors used in prediction are categorical or continuous. As we are going to mention in the following sections, Random Forest is very effective in handling various discrepancies caused by the data structure. We also apply certain optimization methods that allow us to study both location (mean) effect and dispersion (variance) effect of a solution in finding the most preferable solution of the Robust Design problem. Both the Random Forest method and the optimization approaches can be implemented with easily accessible and handy software programs, and they are shown to be effective on example problems. Advantages and disadvantages of the method are explained based on its application on 4 different types of cases.

The main objective of this study is to develop a new robust design method based on Random Forests, for a categorical response variable, which might be an alternative to the one using Logistic Regression. It is not to compare the Logistic Regression or other methods with Random Forest. Accordingly, as the Random Forest method has not been developed for these type of studies, application of it for robust design studies is not straightforward. In order to study performance of this method, a comparison is undertaken with Logistic Regression that is applied to same data sets.

In this thesis, a general review of the RPD and a detailed review of the Random Forest method are presented in Chapter 2. In Chapter 3, the proposed method is described and its applications on our different types of data are reported. In addition, the results are compared with the previous studies and the performance of the method is evaluated. Assessment of all the results obtained are given in Chapter 4. In this chapter, a general evaluation of the proposed method is made, and its advantages and disadvantages are discussed. Finally, in Chapter 5, conclusions are provided about the work carried out and suggestions for further studies are given.

CHAPTER 2

LITERATURE SURVEY AND BACKGROUND

Quality improvement in both industry and service sectors plays an important role in increasing market share. As Pham (2006) states, quality improvement activities during product design and development stages are both efficient and cost effective. Logothesis (1992) also claims that pre-production stages, i.e. designed and analyzed so-called off-line stages, can contribute significantly to the effort to accurately identify and optimize industrial processes, improve product quality, and reduce cost and waste. It is important to improve the quality of product or process at the off-line stages by reducing quality variation based on statistical methods, if possible.

In this chapter, a brief background is provided about Robust Parameter Design, especially for the case of a qualitative response. In addition, a review of the Random Forest method and similar approaches are given.

2.1. Robust Parameter Design

Pham (2006) defines Robust Design as a systematic method to improve the design of a product or process by using statistical techniques. While implementing Robust Parameter Design, we strive to minimize the sensitivity of the product or process to all uncontrollable or noise factors, while keeping the response of the interest on target. Myers et al. (2009) say that, with the Robust Parameter Design methodology, it is aimed to reduce the variance of products or processes by choosing the levels of controllable factors (or parameters) that make the system insensitive (or robust) to changes in a set of uncontrollable factors that represent most of the sources of variability. Since Robust Parameter Design methodology is an off-line approach to improve quality, it is a cost effective approach (Ardakani et al., 2009). The quality response of the interest can be either continuous or categorical. Since continuous responses are considered in most cases, many Robust Parameter Design methods have been developed for such cases. However, categorical response cases are too many to underestimate, yet the studies performed in this area are quite limited.

In industrial processes, categorical responses appear in three different types: binary, ordinal and nominal. In binary response cases, quality characteristic is expressed as pass or fail (or 0-1). Since examining such cases are easier, Robust Parameter Design studies for binary responses are also very common. However, these studies are inadequate to understand and analyze the multilevel categorical response cases we encounter mostly in daily life. In the case of ordinal and nominal responses, quality characteristics assume more than two levels. The ordinal response is a special type of categorical response, the levels of which are naturally ordered. In nominal response cases, the quality characteristics do not have a realistic degree of order, i.e. they cannot be expressed in a certain order.

2.1.1. Robust Design for Categorical Response

One of the important studies about the categorical responses was carried out by Taguchi in 1974. The name of the method is Accumulation Analysis (AA). In AA, cumulative frequencies of categories are analyzed. For each experimental design, frequencies and cumulative frequencies of response categories are determined, then ANOVA is applied to these cumulative frequencies. This method is more suitable for the data that have only categorical controllable factors. In the case of continuous controllable factors, it is necessary to consider only discrete levels of these, which may yield suboptimal solutions. Running AA only for continuous response should be avoided, since considering the discrete levels of continuous data may cause loss of some information (Logothetis, 1992). The method can also be applied to mixed categorical-continuous data. Not allowing the analysis of location and dispersion effects separately is the main disadvantage of the AA method. Also, Box and Jones

(1986) and Nair (1986) criticize the complexity of the method. Nair (1986) proposed a method called Scoring Scheme (SS). In this method, scores are given to the ordered categories and ANOVA is applied to these scores. So, this method is also an ANOVA based method, and the method can analyze location and dispersion effects separately. However, this method has been criticized by some researchers due to its computational difficulties. In addition to that, like AA, this method also cannot be used when the controllable factors are continuous (Erdural, 2006).

Chipman and Hamada (1996) develop the Bayesian Analysis method for categorical response data. For this method, the Gibbs sampling algorithm is used. This method is a strong way to analyze a data set for quality improvement, but the need for complex computer applications makes this approach difficult to use.

Furthermore, Taguchi develops the Weighted Signal-to-Noise Ratio (WSNR), then Wu and Yeh (2006) introduce the method by comparing four different robust parameter design methods. Unlike the traditional SNR, in this method, the weights are given to the categories according to their quality loss. The SNR value which is used to find the optimum levels of the parameters is calculated based on these weights. Like the AA method, location and dispersion effects cannot be analyzed separately by the WSNR method.

Moreover, Jeng and Guo (1996) develop the Weighted Probability Scoring Scheme (WPSS) to overcome the complexity of the Scoring Scheme method. Additionally, Asaiabar and Ghomi (2006) develop the Minimization of Expected Loss (MEL) method, which gives more accurate results compared to AA.

Erdural (2006) proposes a relatively new method, Logistic Regression Model Optimization (LRMO), for categorical response data. In his study, Erdural fits an ordinal logistic regression model to the data, and then estimates the probability of each category for each experimental design point using logistic regression models. For probability estimation, he uses Equations (2.1) and (2.2).

$$\hat{\mathrm{P}}\left(Y_i \le k\right) = \hat{\pi} = \frac{e^{\left(\hat{\beta}_k + \hat{\beta}_1 x_{i1} + \hat{\beta}_2 x_{i2} + \ldots + \hat{\beta}_p x_{ip}\right)}}{1 + e^{\left(\hat{\beta}_k + \hat{\beta}_1 x_{i1} + \hat{\beta}_2 x_{i2} + \ldots + \hat{\beta}_p x_{ip}\right)}} \qquad i=1,\ldots,N,\ k=1,\ldots,K\text{-}1 \qquad (2.1)$$

$$\hat{\mathrm{P}}\left(Y_i = k\right) = \hat{\mathrm{P}}\left(Y_i \le k\right) - \hat{\mathrm{P}}\left(Y_i \le k-1\right) \qquad i=1,\ldots,N,\ k=1,\ldots,K \qquad (2.2)$$

where

x_{ij}: Value of factor j at the i^{th} parameter setting, $j=1,\ldots,p$

$\hat{\beta}_k$: Intercept for the response category k, k=1,…,K

$\hat{\beta}_j$: Estimated coefficient of factor j, $j=1,\ldots,p$

Y_i: Quality response of the i^{th} parameter setting

$\hat{P}(Y_i \le k)$: Estimated probability that the response is less than or equal to category k at the i^{th} parameter setting, $k=1,\ldots,K$

By using these estimated probabilities, Erdural calculates the estimated expected value and variance for each experimental design point i by using Equations (2.3) and (2.4).

$$\hat{\mathrm{E}}\left(Y_i\right) = \sum_{k=1}^{K} k\,\hat{\mathrm{P}}\left(Y_i = k\right), \qquad i=1,\ldots,N \qquad (2.3)$$

$$\hat{\mathrm{V}}\left(Y_i\right) = \hat{\mathrm{E}}\left(Y_i^2\right) - \left(\hat{\mathrm{E}}\left(Y_i\right)\right)^2 = \left[\sum_{k=1}^{K} k^2\,\hat{\mathrm{P}}\left(Y_i = k\right)\right] - \left(\hat{\mathrm{E}}\left(Y_i\right)\right)^2, \qquad i=1,\ldots,N \qquad (2.4)$$

where

$\hat{\mathrm{E}}\left(Y_i\right)$: Estimated expected value of the response at the i^{th} parameter setting

$\hat{\mathrm{V}}\left(Y_i\right)$: Estimated variance of the response at the i^{th} parameter setting

After calculating the estimated expected value and the variance, Erdural (2006) calculates SNR ratios for each experimental design point according to the problem type (smaller-the-better and larger-the-better). For that purpose, he uses Equations (2.5) and (2.6), respectively.

$$\hat{\text{SNR}}_i = -10\log\left(\left[\hat{E}(Y_i)\right]^2 + \hat{V}(Y_i)\right) \tag{2.5}$$

$$\hat{\text{SNR}}_i = -10\log\left[\left(\frac{1}{\hat{E}(Y_i)^2}\right)\times\left(1+3\times\left(\frac{\hat{V}(Y_i)}{\hat{E}(Y_i)^2}\right)\right)\right] \tag{2.6}$$

For both types of the problem, the parameter setting which has the maximum SNR value among those given as output by the full factorial design can be selected as the optimal parameter setting. As the second way to determine the optimal parameter setting, estimated probabilities of the categories can be used. In a full factorial design, the parameter setting that has the maximum estimated probability, calculated by using Equations (2.1) and (2.2), of the desired target category and the minimum variance can be selected as the optimum parameter setting, provided that $\hat{V}(Y_i)$ at the selected setting is reasonably small.

Köksal et al. (2006) arrive at the conclusion that the LRMO method provides an easy and effective way to find the robust parameter setting for the categorical response. This method can be used with both continuous and categorical controllable factors. Furthermore, one of the most important advantages of the method is that we can formulate the RPD problem as an optimization problem and solve it. It can find not only the best design parameter settings among the ones tested in the data collection experiment, but also the best settings that have not been tried before.

Logistic Regression has generally shown a satisfactory performance for tried cases. However, this method has some disadvantages as well, due to problems associated with Logistic Regression modeling, which occur in case of a lot of missing values or imbalanced data. Another disadvantage of the model stems from extremely high regression coefficients in the case of multifactor experiments, especially when the number of factors in the model is close to the number of runs. Random Forest, on the other hand, is shown in the relevant literature to have a better performance than Logistic Regression as mentioned in Section 2.1.1.1.

Lastly, Karabulut (2013) compares the methods, LRMO, AA, WSNR, SS, and WPSS for five different cases in her M. Sc. thesis study. She finds that LRMO shows the best solution performance according to the performance measure which is calculated depending on Logistic Regression for all of the selected examples, and AA shows the best performance according to the performance measure depending on ANOVA for all the examples. In addition to LRMO and AA, the WSNR method also shows a satisfactory performance, and the WSNR method is rather practical. For all methods, factors are treated as categorical, but Logistic Regression allows analyzing also the factors as continuous. Karabulut (2013) has commented that, due to this property, had the appropriate factors been set as continuous, the method would yield even better results.

For robust design, Köksal et al. (2006) and Karabulut (2013) claim that Logistic Regression is one of the best methods for a categorical response case. Nevertheless, there are also cases where Logistic Regression is not adequate. We mention these deficiencies in Section 2.1.1.1 Random Forest, on the other hand, may overcome these problems, which are introduced in Section 2.2.1.

2.1.1.1. Comparing Random Forest with Logistic Regression

Erdural (2006) uses Logistic Regression method as a classifier in Robust Parameter Design for a categorical response. We, on the other hand, use Random Forest. Comparison of these two classifiers has been performed by many researchers in the past. Almost all of them defend that the Random Forest algorithm gives better results (in terms of many criteria like accuracy, sensitivity, Area Under the Receiver

Operating Characteristics (ROC) curve (AUC)). Lin et al. (2004) compare LR and Bayesian Network approaches with Random Forest by using the ROC curve as the performance criterion, and they observe that Random Forest shows better performance over both. Yoo et al. (2012) report that while using Logistic Regression is not recommended, if the number of observations is less than the number of the attributes (because of the degrees of freedom), Random Forest can be used with many more variables than the number of observations. In addition to these, Lee et al. (2005) emphasize that Random Forest always performs much better than Logistic Regression and CART in all of their seven data sets.

Logistic Regression is a very popular algorithm and Random Forest is a relatively new algorithm. Each one has some advantages over the other. If we want to compare both methods; firstly, Random Forest can handle the missing value problem automatically (details of the method are explained under Section 2.2.1.5). However, Logistic Regression tries to overcome this problem by applying stepwise variable selection and in this process, only significance levels of the attributes/variables are taken into account over many hypothesis tests. But this may result in a decrease in the predicted power. Furthermore, when data contain a large number of variables/attributes, overfitting may occur with the Logistic Regression algorithm, but Random Forest almost never encounters an overfitting problem (Geng, 2006). Logistic Regression, on the other hand, is a parametric method based on some distribution assumptions. This is an advantage for making statistical inferences, but a disadvantage if assumptions are not satisfied. Unlike Logistic Regression, Random Forest does not require such assumptions as a non-parametric method. Moreover, Random Forest can automatically handle imbalanced data by giving more weight to the classes that contain a smallar number of observations. This property is favorable especially when the number of observations in the class of interest is small. However, Logistic Regression shows poor performance in such a situation, because it cannot handle this problem by itself (Geng, 2006). In addition to these, Geng (2006) reports that compared to Logistic Regression, Random Forest performs better in small data, because Random Forest does not require splitting to train and test sets, but it makes its own cross validation on its own. For Logistic Regression to make cross validation, a test set is needed. Moreover, Random Forest calculates proximities between pairs. It makes easy to understand which observations are similar to each other and show similar behavior. Logistic Regression does not calculate proximities between any pairs.

Geng (2006) specifies that when Logistic Regression and Random Forest are compared, each method has some superior properties to the other one. According to the results of Geng's study, while the Random Forest method is successful at handling the missing value, Logistic Regression shows better performance in terms of correctly classified rate for the data set Geng used. Furthermore, Yoo et al. (2012) emphasize that Logistic Regression is the most popular method for analyzing the relationship between multiple design parameters and categorical response. Also, Muchlinski et al. (2016) conclude that if the data set has a linear relationship between the input and output, and the data is relatively balanced, Logistic Regression will perform very well. When the relationship between the response and the predictor variables is truly linear, then Random Forest will only approximate the Logistic Regression model. According to Geng (2006), the most important advantage of the Logistic Regression is that odds ratios of each design parameter and its confidence intervals, which cannot be achieved in Random Forest, can be calculated automatically, so that it can be determined how factors affect the response. We can summarize all these comparisons as in Table 2.1.

2.1.2. Optimization of Robust Design Problems

Robust product or process design is an important approach for achieving a high quality at a low cost (Phadke and Dehnad, 1988). It can make a product or process insensitive to noise (uncontrollable) factors, while achieving quality targets. So, during the product and process design, we have two goals: the first one is to achieve the desired target value, and the second one is to minimize the variation. For the optimization of the product or process design, we use two different methods in this study. One of them is a non-linear multi-objective optimization method; the other one is the maximization of SNR method.

Table 2.1. Comparison of Random Forest (RF) and Logistic Regression (LR)

Features	**RF**	**LR**	**References**
Automatically handling missing values	+	-	Geng (2006) Muchlinski et al. (2015)
Automatically determining importance levels of variables	+	-	Geng (2006) Ruiz-Gazen and Villa (2007) Yoo et al. (2012)
Distributional assumptions	+	-	Muchlinski et al. (2015) Geng (2006) Ruiz-Gazen and Villa (2007) Yoo et al. (2012)
Overfitting	+	-	Geng (2006) Ruiz-Gazen and Villa (2007)
Handling imbalanced data	+	-	Muchlinski et al. (2015) Geng (1992) Ruiz-Gazen and Villa (2007)
Good interpretation with small data	+	-	Geng (2006)
Probabilistic output	-	+	Geng (2006)
Calculating proximities between pairs	+	-	Geng (2006) Muchlinski et al. (2015)
Ease of interpretation	-	+	Geng (2006)
Classification of multiple classes	+	+	Yoo et al. (2012) Muchlinski et al. (2015)
Calculating confidence intervals	-	+	Geng (2006)

2.1.2.1. Non-linear multi-objective optimization method

The aim of Robust Design studies is minimizing the variance of the quality characteristics, while achieving the desired quality characteristic at the same time. For that, there can be more than one objective in the Robust Design problem such as these. These objectives can be expressed mathematically using empirical models of the quality characteristic mean and variance in terms of setting of the design parameters.

Therefore, the Robust Design problem can be solved as a multi objective optimization problem. There are lots of methods for solving multi-objective problems. In our study, we use ε-Constraint Method, which is one of the most commonly used methods for multi-objective optimization problems. The working procedure of the ε-Constraint Method is described below:

ε-Constraint Method: This method is developed by Haimes et al. (1971). In this method, an objective is selected out of a set of objectives and optimized, and the other objectives are converted into constraints by selecting an upper/lower bound for each of them (Miettinen, 2012). Thus, the model is transformed into a single objective model, and the solution can be obtained by a suitable method and software. So the mathematical models of the problems turn into:

- For minimization problems:

$$\begin{array}{lcl} Min\ z_i\ (x) & & Min\ z_i\ (x) \\ Min\ z_j\ (x) & \longrightarrow & s.t. \\ s.t. & & z_j\ (x) \leq \varepsilon \quad j \neq i \\ \mathrm{x} \in \mathrm{S} & & \mathrm{x} \in \mathrm{S} \end{array}$$

where ε is the upper bound.

- For maximization problems:

$Max\ z_i\ (x)$		$Max\ z_i\ (x)$
$Max\ z_j\ (x)$	$\longrightarrow$	$s.t.$
$s.t.$		$z_j\ (x) \geq \varepsilon \quad j \neq i$
$x \in S$		$x \in S$

where ε is the lower bound.

To obtain more robust results and avoid favoring in the tradeoff between any objectives, the summation of all epsilons (used as both lower and upper bound) can be added to the objective function. In our study, we have only two objectives. One of them is selected and optimized and the other one is turned to a constraint, resulting in only one epsilon value. Correspondingly, we add the objective which is turned to constraint to the objective function after multiplying it with a very small coefficient instead of adding the summation of epsilon.

2.1.2.2. SNR method

For this method, unlike the non-linear multi-objective optimization method, expected value and variance are combined into a single performance criterion. The name of the performance criterion is Signal-to-Noise Ratio (SNR). The SNR is calculated by using the expected value and variance. Depending on the type of problem, the SNR formula also changes. These formulas are given in Table 2.2, for μ: mean, and σ: variance of a response.

Table 2.2. SNR Formulas for Different Problem Types

Problem Types	SNR
Smaller-the-Better	$-10\log_{10}\left(\hat{\mu}^2+\hat{\sigma}\right)$
Larger-the-Better	$-10\log_{10}\left[\left(\frac{1}{\hat{\mu}^2}\right)\times\left(1+3\times\left(\frac{\hat{\sigma}}{\hat{\mu}^2}\right)\right)\right]$

Firstly, for an experimental design point, probabilities of all categories are obtained, separately, and, then, by using these probabilities, the expected value and the variance are estimated. Then, utilizing the expected value and the variance, the SNR is calculated for each experimental design point. For the optimal parameter setting, the design that has the maximum SNR value is chosen for both smaller-the-better and larger-the-better types of problems using ANOVA.

As we mention before, we find the Random Forest method successful in coping with situations where LR might be insufficient. The Random Forest method is a special case of ensemble techniques for decision trees only. We present ensemble techniques and the Random Forest method, in the following.

2.2. Ensemble Techniques

Ensemble is a method, which has been frequently used in Data Mining and Machine Learning recently, and which improves the accuracy of individual classification methods by combining them (Eldardiry and Neville, 2011). This method also decreases the classification errors of individual methods. Classification error contains two components: bias and variance. Bias is the tendency of a classifier to estimate (overestimate or underestimate), and variance is the sensitivity shown by the classifier over the different learning data sets (Sun et al., 2007). There is a trade-off between these components. Individual methods can have a low bias and, at the same time, they might have a high variance. Having high variance means that being not a sufficiently good predictor. In this situation, the combination process decreases the variance. Hence, the aim of the ensemble is that while it fixes bias, at the same time, it decreases variance by combining such individual methods.

According to Okun and Valentini (2009), ensemble can be carried out in different forms based on used models and used data set. One of these, just one classifying method is applied to different subsets of a data set. That is called a homogeneous model. Applying different classifying methods to a data set is another ensemble form, and that is called a heterogeneous model.

The main idea of the ensemble methodology is to combine many individual models to get a classifier model that outperforms every one of these, and also to avoid the overfitting that can be seen in a single model.

For the classifiers that have just the class label as output, votes which are given for each class by each classifier in the ensemble are determined, and the highest voted class is determined as the class of the ensemble method that is voting principle for an unseen data (Rokach, 2010). Or weights are given according to the performance of each classifier, and votes are evaluated based on these weights. Therefore, sometimes, performance of a classifier is better than the other classifiers, and in this case, the weight of this better classifier would be higher than the others (Zhou, 2012). For the classifiers that have a probability as the output (like Logistic Regression), the results of each classifier in the ensemble are combined by weighted or regular averaging that is probabilistic principle, and the final result becomes the result of the ensemble (Rokach, 2010).

There are two main ensemble approaches: Bagging and Boosting. And also, there is a special form of bagging, Random Forest, which is focused on the most in this study.

Bagging (Bootstrap Aggregation)

Bagging (Breiman, 1996), whose name comes from a blending of the words “bootstrap” and “aggregation”, is based on two main procedures: Firstly, a desired number of bootstrap samples and an aggregation of these are generated. Then, the result of this aggregation is combined to get the result of the bagging. Bagging is a simple and effective ensemble method. Its application is straightforward: new data sets are created randomly from the original data set with replacement (that is each observation in the original data is equally likely to be sampled each time an

observation is requested) and at the size of N, where N is the size of the original data set, and results are combined according to outputs (voting or averaging, etc.). A pseudocode of Bagging algorithm, which is described by Rokach (2010), is given in Appendix A.
Since each new train set, which is constructed at size N, is randomly generated with replacement, many experiments (In data mining literature, these are called 'data points' while, in the design of experiment literature, as 'experimental run'. In this study, we use it as experimental run or experiment, interchangeably) are repeated in the same subset, while some of the experiments are never seen in a subset. However, since each subset is created randomly and many subsets are generated, the probability for each experiment to exist in any subset is quite high.

Bagging has a better performance than each individual classifier in itself, as the main goal of the ensemble (Breiman, 1996). It reduces the variance of each single model by a combining procedure. Therefore, classification error is also reduced. In fact, Breiman (1996) observes that, when bagging was tested on real and simulated data sets, accuracy was improved prominently both for regression and classification.

Boosting

Boosting was developed by Yoav Freund and Robert E. Schapire in 1996. It is an ensemble method which turns a weak classifier into a powerful one. The working principle of the Boosting algorithm is based on the principle of step-wise correction of the weak classifier. In the algorithm, classifiers are dependent on each other. So, the observations which were classified incorrectly by the previous classifier are more important for the next classifier. The working procedure of Boosting is a bit more complicated than that of the bagging algorithm. The algorithm works as follows: Initially, a subset is randomly created from the original data set without replacement. After that, this subset is trained on a classifier. The experiments which were wrongly classified by the first classifier must definitely be in the subset that is created for the second classifier. And remaining part of the data in the second subset are selected from the original data set, except the experiment that has already been used by the first subset. This new subset is trained on the second classifier, and incorrectly classified observations are used for the third subset. And the process is repeated until a pre-determined stopping criterion is met. This stopping criterion can be a number of iterations or a fixed point in time beyond which accuracy does not increase anymore, etc. At the end of the algorithm, the class that has the majority of votes is determined as the class of the Boosting.

AdaBoost

AdaBoost is a powerful adaptation of boosting. The main aim of the AdaBoost algorithm is to focus on incorrectly classified observations. Unlike Boosting, here, weights are given to observations, and focusing is determined according to these weights. Initially, all observtions have equal weights (1/N, where N is the number of observations). At each iteration, weights of incorrectly classified observations are increased, while weights of the correctly classified observations are decreased. The process continues in this manner by updating weights each time. Note that equal weights are assigned to all incorrectly classified observations for the next classifier. Aside from that, weights are also given to each classifier according to their accuracy. In this way, after the process is stopped according to desired stopping criteria, weights of the classifier are multiplied by the outputs of the classifier (vote or probability), and weighted results are obtained.

Boosting algorithm may fail at overfitting. To be able to avoid the overfitting, the number of iterations should be set as small as possible (Rokach, 2010).

We compare the bagging and boosting algorithms as summarized in Table 2.3.

Table 2.3. Comparison of the Bagging and Boosting

BAGGING	**BOOSTING**
Bagging is a random operation	Boosting is an adaptive operation
Bagging reduces the variance	Boosting reduces both variance and variance (but variance may increase when the classifier is stable, like logistic regression, etc.)
Thanks to randomization, it is fast	Due to reweighting, it is slower
Bagging can work in parallel form (simultaneously)	Boosting is a consecutive method, so it cannot work in parallel form
Probability of selecting of each observation for a subset in bagging is equal	Observations are selected according to their weights in boosting

2.2.1 Random Forest as a Type of Bagging

Bagging and boosting, as mentioned in Section 2.2, are ensemble techniques for reducing the variance of an estimated prediction function. Random Forest (or Random Forests) (Breiman, 2001) is an important modification of the bagging techniques that construct a large number of de-correlated decision trees and combine them (Hastie et al., 2008). The combination means taking the average of these for regression and letting them vote for selecting the most popular class for classification. Hastie et al. (2008) explained that the idea of Random Forest is to improve the variance reduction of bagging by reducing the correlation between the trees, without increasing the variance too much.

In Random Forest, firstly, data must be analyzed by selecting a target class and independent variables. Once the target and features are determined, Random Forest begins by growing a decision tree to the maximum possible size (Breiman et al. 2012). Random Forest-trees differ from standard decision trees in several important ways.

Firstly, Random Forest-trees do not use all the available observations in the data set for growing the trees. Instead of that, it uses a bootstrap sample that generally includes two thirds of the original data set (Geng, 2006, Breiman et al., 2012). This ratio may change according to the data set and used software. But it can be difficult to choose the best model among different models that come out when this ratio changes. This ratio is also proposed by Breiman (1996), who developed the Random Forest. A bootstrap sample typically has the same number of records as the original data set, but 37% of original data records are not included in the bootstrap sample. To fill in the size of bootstrap to N (the size of the original data set), the 63% of the records are selected more than once. This method is called "sampling with replacement" (Breiman, 2001). Each experiment has an equal chance to be within a bootstrap sample every time. During this process, every experiments has the possibility to be drawn. So, the reason why each observation is not found via bootstrap samples prepared for each tree is because the selection is carried out randomly and with replacement, and it allows some of the observations to be selected more than once. It does not matter to us whether a record has been drawn in this process, which is why we can see the same records being drawn more than once and others not at all.

The second difference is in the splitting part. While growing a tree in a Random Forest, selection of a split variable (feature, attribute or controllable factor) in a node is carried out partly or wholly random. The number of the variables for splitting at each node always must be between 1 and the number of variables. Usually, the number of the randomly selected variable is the integer part of the square root of p, as suggested by Hastie et al. (2009), where p is the number of features. Moreover, Breiman (2001 and 2003) suggests the number of the randomly selected variables for splitting would be as well one-half or twice the square root of p, $\lfloor \sqrt{p}/2 \rfloor$ and $\lfloor 2 \times \sqrt{p} \rfloor$, even $\lfloor (\log_2 p) + 1 \rfloor$, depending on the data set. Furthermore, recommended number of variables to split for regression is $\lfloor p/3 \rfloor$ (Hastie et al. 2008). This random splitting process provides that the model be a form of true Random Forest instead of a pure bagging. After determining the number of eligible predictors for the node, these numbers of variables are selected randomly. Then the best splitter is chosen from among these eligible predictors as in a normal decision tree. Once the data is split, this process will be repeated for each node until the growing of the tree is stopped.

Finally, decision trees in the Random Forest are grown as large as possible, and left unpruned. In appearance, growing a large tree at random and then leaving it unpruned is not a reasonable approach to develop a satisfactory model, ‘in fact, it is not’, particularly since it results in overfitting. However, since this process is repeated for hundreds or even thousands of times, and the results of these large number of trees are combined, more powerful results would be obtained compared with non-random methods; such as a solo decision tree, logistic regression, support vector machine or neural network.

A question that might come to mind is how this process can yield strong results. The key insight is that the individual models tend to be good standalone models, and can even build the best possible single models. However, Random Forest method follows a different way; according to Random Forest, weaker stand-alone models combine more expertly than the powerful standalone model. The first reason for this is when powerful individual models are grown, these models are too similar to each other; even in the extreme case, all of them are the same. In this case, combining or averaging of these similar models do not provide any benefits. The ensemble model also would be similar to the individual models. However, it is clear that if the individual models are different, it is meaningful to combine them since it would provide benefits in terms of accuracy. Selecting a different subset while growing each individual tree, and also selecting a different variable for the splitting (two-parts randomization) of all nodes provide the individual models (trees in Random Forest) to be quite different from each other, and make the correlation between the trees minimum. The other reason for using weak models to combine is that it would be a form of “slow learning”, which provides considerable benefits regarding the prediction error of the fitted models, as discussed by Freidman et al. (2004). The third reason why it is better to use weak models is that they have low bias and high variance, and by combining them, low variance is also obtained. At the end of the process, a model with a low bias and a low variance is achieved.

Afterwards, the data to be classified is passed down to these all unpruned trees. The class which is mostly chosen by trees in the forest is determined as the class of the data.

In regression, the situation is different: here, after the data is passed down to the trees, the average (or weighted average) of the results of the trees are calculated, and the value of the data is found. The algorithm of Random Forest for regression and classification, which is written by Hastie et al. (2008), is presented in Appendix A. And also an illustration of Random Forest represented by Machado et al. (2015) is given in Figure 2.1.

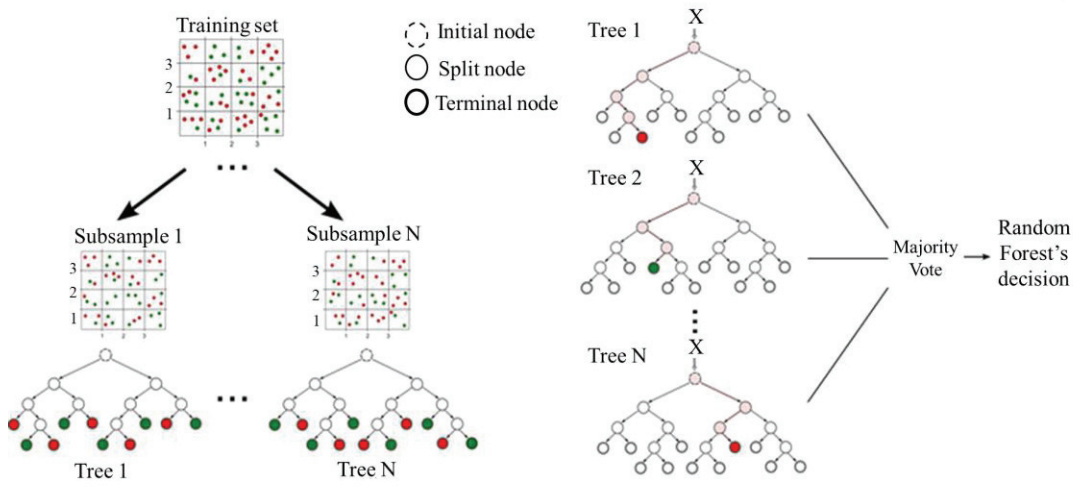


Figure 2.1 Illustration of Random Forest for Classification
(Source: Machado et al. 2015)

2.2.1.1. Advantages and Disadvantages of Random Forest

Random Forest is an effective method for prediction. In addition to this, some of many advantages of the Random Forest method (compared to the existing methods) are:

- Since decision trees are nonparametric methods, Random Forest, too, is a non-parametric method. So, the data do not need to be transformed, modified, or rescaled.
- Handles the missing values automatically (by estimating), even if large proportions of the data are missing.
- Random Forest models are usually more accurate than a single decision tree and many other algorithm currently available. In addition to this, Random Forest predicts unbiased estimates.
- Since each tree in the forest is different from each other, growing large numbers of trees with double randomization tends to remove overfitting.
- Random Forest has a self-testing procedure; that is, Out-of-Bag (OOB) Error, which is a form of cross validation, and it provides a deeply reliable evaluation in Random Forest. Due to this self-testing for OOB, there is no need to split data into train and test sets.
- Random Forest estimates which variable is important for the Random Forest model by using Gini Index and Accuracy.
- Random Forest runs very effectively on large data sets, and trees are grown very fast (because a small number of variables is used for each time-double randomization procedure).
- Random Forest computes the proximity matrix that shows the proportion of how often two experiments (data points/observations) end in the same leaf node for different trees. It shows the similarity between a pair observation. These proximities might also be used in clustering.

Random Forest also has some minor disadvantages that can be ignored beside its advantages.

- Random Forest is a data hungry method like the decision tree it is based on. So, Random Forest offers a better solution when the size of the data is large, although it shows relatively better performance on small data sets than many other methods.

- Constructed trees for the forest are never known, so it is hard to understand as to what the Random Forest really does. The algorithm is an incomprehensible "black box".
- In the Random Forest method, some arguments change based on the data type and also need to be decided by the user. These are:
  - <u>Number of the split value:</u> This value is related to parameters used while splitting each node. To determine the optimal value of split value, Breiman (2001) suggests a general rule, but he also says that this rule can change according to the data set used. Using a lower split value provides less correlation between trees, but, at the same time, decreases the strength of each tree. An optimization between these is needed.
  - <u>Number of trees:</u> Another parameter to be decided is the number of trees used in the forest. This argument also changes depending on the data set. A small number of trees decreases the strength of the forest, on the other hand, using a very large number of trees means growing similar trees, and it also takes too much time.
  - <u>Weights:</u> The most difficult argument to decide on is the weights to be given to the classes. We can grow a Random Forest without giving weights to classes, but to obtain more robust results, especially in imbalanced data sets, using this property of the Random Forest method increases the performance of the model. There is no strategy for giving weights. It depends only on the decision maker and her/his purpose.

2.2.1.2. Out-of-Bag samples and errors

Out-of-Bag (OOB) is one of the most important features of the Random Forest method, and it is also specific to models utilizing bootstrap aggregating only. OOB is used to understand Random Forest correctly. When starting to grow a Random Forest tree, not all of the learning data is used. Firstly, a special sampling method is used to create new learning data, i.e. a bootstrap sample. Random Forest constructs new bootstrap samples for each tree separately and independently. Owing to this and to replacement strategies, bootstrap samples of trees are never exactly the same, even though each bootstrap sample is created by using the same original data set. After decision trees are grown and Random Forest is completed, OOB samples are used as test data to measure the performance and predictive accuracy of the Random Forest results. While testing each OOB sample, the trees which do not contain the related experiment are used. This process continues until all OOB samples are tested. The proportion of the wrong classification of these OOB samples in constructed Random Forest gives the OOB error rate of the Random Forest. Breiman (2001) suggests that one-third of the original data set is used for OOB sample, and the remaining two-thirds of the data set is used to generate bootstrap samples.

One of the benefits of the OOB is that since it splits the data as train and test sets by itself, there is no need to split the data before starting the algorithm. This property makes Random Forest quite convenient and effective, especially for analyzing small data sets, since the data which is already small would not be smaller. With this feature, we can call the OOB a superior version of the cross validation (Breiman et al., 2012).

Generally, the greater the number of trees, the more reliable the Random Forest method becomes. When each tree in the forest is grown, a new OOB sample is created; by means of that, for each time, a different experiment is used for testing. It brings about an increase in the reliability of the method.

2.2.1.3. Importance of variable

During the analysis of a data set, especially in cases with a large number of variables, knowing which variable has what kind of an impact -and to what extent- helps us to understand the data set under consideration. Focusing on a certain number of variables, instead of a large number of them, simplifies the analyses of data sets.

Random Forest uses two different methods to determine the importance of a variable in a data set. For one of them, as in the other tree-based models, the importance is measured by the role a variable plays during model construction. This is called Gini-based importance, and Breiman et al. (2012) explain it as "*technically, variable importance is based on how often a variable is used in a tree (or collection of trees), what fraction of the data passes through a given node being split, and how well the splitter performs in a node as it separates levels of the target variable.*" In a single-tree model (like CART), variables are examined just once. However, in a Random Forest model, the score of a variable is computed for every split the variable generates. After all scores are summed, scores are normalized by assigning 100 as the highest score. The most important characteristic of Gini-based variable importance is that it shows the role of the variable, while Random Forest model is being constructed.

Random Forest also uses another enthralling and original method, accuracy based variable importance (Breiman et al, 2012). This method runs as follows: firstly, the random forest model is generated, and the accuracy of the forest is calculated. Then, all column values of a variable in the training set are "randomly" scrambled; there are no changes introduced for the other variables. This new design of the training set is passed through the tree, and a new accuracy is calculated. This process continues by re-shuffling the same variable for each tree, until no tree that is not passed down in the forest. Then, the accuracy obtained from the original data (without shuffling) is compared with the one obtained from reshuffling the data. If the accuracy decreases significantly, we would consider that the variable is important; on the contrary, if there is no significant change in the accuracy, we would consider that the variable is insignificant. This process is followed for all variables, and the degree of importance of each one of these is determined. By using this property, we can actually see which variables affect the performance of the Random Forest more or less. In other words, we can observe what would happen when the model is constructed without the variable in question. One of the charms of the method is that, although each variable is re-shuffled for a number of times equal to the number of created trees (for example; if we have N tree and M variable, the re-shuffling process is carried out for NxM times), the process moves fast. We can see an exemplary variable importance plot (created by using R-studio software) for both measures (Gini-based and Accuracy based), in Figure 2.2.

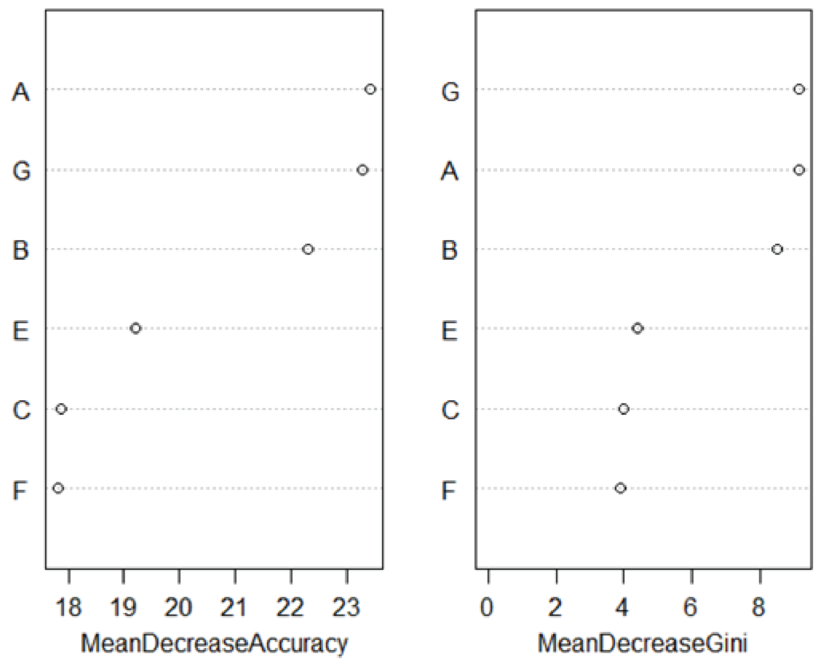


Figure 2.2. An Example Variable Importance Plot

2.2.1.4. Proximity matrix and plots

Proximity matrices and proximity plots are the most creative outputs of the Random Forest method. Proximity measures similarity between a pair of observations. While doing so, it considers how often the two observation is located in the same terminal node. Proximity works as follows: after a tree in the Random Forest is grown, all data (both train and OOB data) are passed through the tree, and pair of observations that are in the same terminal node are determined. For each of the two observation set in data, this process is repeated until the last tree in the forest. Then, the number of trees that were in the same terminal node for each pair is summed, and then this summation is divided by the number of trees in the Random Forest. Thereby, each cell in the proximity matrix (i,j) shows the proportion of trees for which i and j are located on the same terminal node. It is obvious that a higher ratio shows i and j are proximate.

Hence, the maximum number of trees for which a pair of observations is ended up in the same terminal node is equal to the number of the trees in the forest, and naturally, the minimum value is 0.

Since Random Forests consist of trees of maximum size that cannot be split anymore; i.e. unpruned trees, their terminal nodes are pure, which means that there might possibly be only one single observation in many terminal nodes. For that reason, if the observations do not have many features in common, the probability of this pair of observations to be within the same terminal node is very low. So, if a certain pair of observations is located at the same terminal node among these unpruned trees, their many, or even, all features are expected to be the same, and the proximity of these would be equal to 1. On the other hand, if any two observations never come together, they are probably very different from each other, and the proximity of these would be equal to 0.

For a data set which has N number of observations, a NxN proximity matrix is accumulated. Therefore, as the number of samples grows, the size of the matrix grows exponentially. When the matrix is too large, it becomes difficult and meaningless to examine it. Breiman and Cutler (2012) examine how large this matrix is, as shown in Table 2.4.

Table 2.4. The Size of the Proximity Matrices

Number of Observation in the Data (size N)	**Number of Cells in the Proximity Matrix (NxN)**	**Size**
100	10,000	40 KB
1,000	1,000,000	4 MB
10,000	100,000,000	400 MB
100,000	10,000,000,000	40 GB
1,000,000	1,000,000,000,000	4 TB

From Table 2.4, it is clear that working with the full proximity matrix with more than 10,000 observations is not practical, and even impossible for many environments. For data sets with more than 10,000 observations, it would be more logical to use proximity plots.

Proximity plots are drawn by using proximity matrices. In Figure 2.3, there is an example proximity plot created by Quach (2012). Proximity plots show similarities and distances between the observations. Hastie et al. (2008) point out that proximity plots give an indication of which observations are effectively close together in the eyes of the Random Forest classifier. This property also provides convenience, especially for clustering data sets. Furthermore, proximities are used for filling missing values in data sets as mentioned in Section 2.2.1.5.

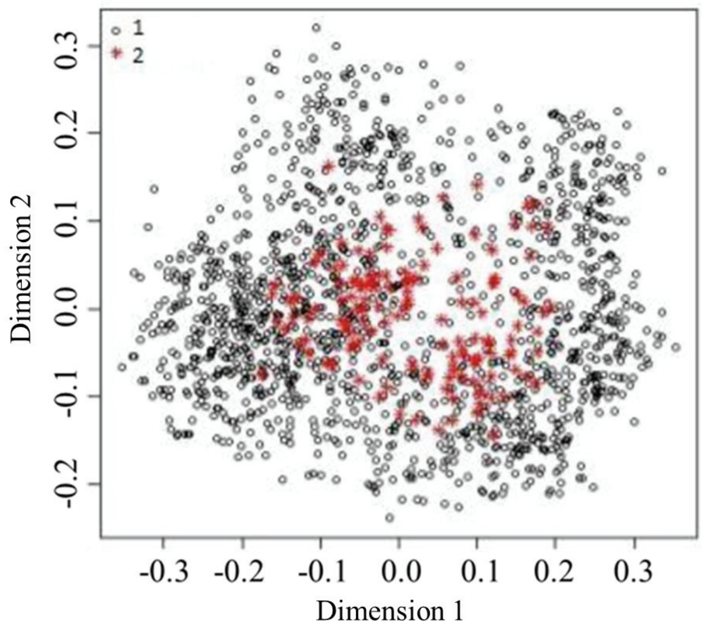


Figure 2.3. A sample Proximity Plot (Source: Quach, A. T., 2012)

2.2.1.5. Missing value handling in Random Forest

During data collection, a great amount of data may be censored, or some missing values may exist due to recording errors. These missing values cause some problems in statistical analyses, because a lack of information decreases the accuracy of the analysis. There are different ways which Random Forest can use to impute missing data. He (2006) lists these approaches in his Ph.D. thesis as follows:

> "*1. For numerical variables, NAs are replaced with column medians.*
> *2. For categorical variables, NAs are replaced with the most frequent levels (breaking ties at random).*"

Random Forest also offers an advanced method for handling missing value problems. Missing cells are filled by using values that are obtained from the matrix of proximities (He, 2006).

If an observation has continuous-type missing values in its variables, these missing values are filled with the weighted average of the observations that do not have any missing value. Here, weights are assigned according to proximities between the observation in question and the other observations. So, the weight of an observation is calculated based on the proximity of the observation to the observation in question. If an observation, on the other hand, has categorical-type missing values, these empty cells are filled with the most frequent non-missing value of the related variable, where frequencies are weighted by proximities. This process is repeated with newly filled cells. For each repetition, all empty cells are filled by the new proximities calculated with newly filled cells. The process continues until all empty cells are filled. According to Breiman and Cutler (2017), usually, repeating this process for 4-6 times would be sufficient. Automatically doing this process makes Random Forest a preferable method. In addition to this, even if there is a large number of missing values, Random Forest can handle these as well. Breiman and Cutler support this information with their study of DNA processes. According to them, more interestingly, even when 50% of the DNA data were deleted, test set error rate was still less than 10%.

When missing values are encountered in a test set, two different methods are applied for imputing depending on whether labels exist or not (Breiman and Cutler, 2017). If the missing value has a label, empty cells are filled by using the training set. If the missing value does not have a label, on the other hand, the related observation is replicated for a number of times equal to the total number of classes. Then, each replicate is matched with class labels one at a time. The obtained test set is passed down the Random Forest, then the class receiving the majority of votes is admitted as the class of the observation.

2.2.1.6. Random Forest prediction and performance measures

Random Forest makes predictions via voting and averaging for classification and regression, respectively. In other words, rather than a single Random Forest-tree making a prediction, each terminal node votes for one class for an unseen data. According to this, the class that receives the most votes is determined as the class of this unseen data.

In Random Forest, a single tree does not have a prediction, because trees in the forest are grown to the maximum possible size, and, for that reason, a terminal node generally includes only one observation, and generating a prediction with only one observation is not suitable. Instead, the common decision of trees is taken into account. The prediction is made by calculating the rate of votes given for each class. Since double randomization (selection of learning data-bootstrap sample and selection of variable for splitting) process is carried out while growing a Random Forest, there is voting variability. However, despite this double random growing, if the method chooses a class as the class of an observation, it is a powerful indicator that this class might indeed be the right class.

For performance measures of the Random Forest method, there are many different alternative methods, the most of which are also used by other data mining algorithms. The OOB error rate graph can be used as a visual output of the performance of Random

Forest method. In addition, as in the other machine learning methods, the confusion matrix is obtained, and in parallel with this matrix, Accuracy, Misclassification rate, Kappa, F-Measure, G-Mean, Recall, Precision, and Specificity can be calculated. Also, AUC (The Area Under the Receiver Operating Characteristic (ROC) Curve) can be calculated. Literature offers a number of performance measures, and some of these used in our study are given in Appendix B.

Furthermore, there is another important measure, balanced accuracy. As Brodersen et al. (2010) said balanced accuracy is used when data is imbalanced. By using balanced accuracy, we overcome the possibility of obtaining any wrong indicators as the result of imbalancing. For that reason, we calculate both accuracy and balanced accuracy for the example problem.

Let us consider the confusion matrix below:

Table 2.5. An Example Confusion Matrix

		Predicted					Sum
		Class I	Class II	Class III	…	Class K	
Actual	Class I	a					A
	Class II		b				B
	Class III			c			C
	⋮						:
	Class K					m	M
	Sum	X	Y	Z	…	T	N

For the equally weighted classes, balanced accuracy formula (Bordersen et. al., 2010) is as follows:

$$\text{Balanced Accuracy} = \frac{1}{K}\left(\frac{a}{A} + \frac{b}{B} + \frac{c}{C} + \ldots + \frac{m}{M}\right).$$

Let w_i be the weight (the cost associated with the misclassification) of Class *i.* Then, Bordersen et. al. (2010) suggest that the weighted balanced accuracy is calculated by:

$$\text{Weighted Balanced Accuracy}=\left(w_1\times\frac{a}{A}\right)+\left(w_2\times\frac{b}{B}\right)+\left(w_3\times\frac{c}{C}\right)+\ldots+\left(w_K\times\frac{m}{M}\right)$$

2.2.1.7. Random Forests and overfitting

Overfitting is an interesting subject for the Random Forest method. Although Random Forest itself generally cannot overfit, almost all Random Forest-trees are overfitted to the bootstrap in which they are grown (Breiman et al., 2017). But, we are concerned with the performance of Random Forests on unseen data, not the performance of each individual Random Forest-tree. Many studies show that, even though the performance of a single Random Forest-tree is poor, the performance of Random Forest formed by the ensemble of these individual trees is very powerful on new data (Breiman, 1996). As we mentioned in Chapter 2.2.1, while estimating a class of an unseen data, in cases where single Random Forest-trees are weak, Random Forest performs better compared to the cases where single Random Forest-trees are strong. Our aim to create an ensemble is to get the model which has a low bias and no unnecessary variance. But, when we prune the tree to prevent the overfitting, bias increases and we cannot eliminate this even if we resort to creating an ensemble (Breiman et al., 2012).

Surprisingly, it is expected that when the number of growing trees in the forest increase, Random Forest can overfit, because of the risk of similarity among grown trees. However, double randomization and averaging process remove this risk.

2.2.1.8. Random Forests for regression

In classification, the response of the data is a class label (categorical); in regression, on the other hand, the response is a numerical value. Therefore, while prediction is made by voting in classification, the prediction in regression is obtained from the simple or weighted average of each tree (Hastie et al., 2008). Another difference between classification and regression in the Random Forest method is the number of the splitting value used in the method. There are different formulas for regression and classification to determine the splitting value to be used in a node as mentioned in Section 2.2.1. Furthermore, Breiman also pointed out in his study in 1996 that Random Forest performs better for regression compared with classification.

CHAPTER 3

THE PROPOSED METHOD

The proposed method for RPD for the case of an ordinal response variable is described in this chapter. Applications of the proposed method on different data sets are also given and discussed. Moreover, in this chapter, comparison of our proposed method and many other methods that have already been applied on the data sets is made, and the strengths and weaknesses of the proposed method are examined based on these applications. Three examples are chosen towards this end. One of them are smaller-the -better type problems, and the two others are larger-the-better type problems. In all of them, responses have multi-class outputs.

Many studies associated with the RPD problem have been performed, but the studies regarding the problem that has a categorical response are limited. Among the researchers in this area; Erdural (2006) studies this type of problem and he uses the Logistic Regression as the solution method, and Karabulut (2013) analyzes five different cases by using five different methods; namely LRMO, AA, WSNR, SS, and WPSS methods (see Section 2.1.1 for details). Comparing the methods, Karabulut (2013) has chosen the design parameters as categorical since methods she used are ANOVA based. However, some parameters; such as temperature or pressure can be treated as continuous, in order to obtain more robust results.

Logistic Regression, which is used by both Erdural (2006) and Karabulut (2013), is a very powerful method in many respects. Logistic Regression is powerful not only for prediction, but also for analyzing both categorical and continuous design parameters. However, there are some cases where Logistic Regression fails, too (see Section 2.1.1.1). That is why we consider the use of more powerful nonparametric methods, and we find the Random Forest method to be appropriate due to the properties of it that are claimed to be superior to Logistic Regression as discussed in Section 2.1.1.1 in the literature. A flowchart of the proposed algorithm based on Random Forest as a classification method is provided in Figure 3.1. In the sequel, we explain our algorithm step by step.

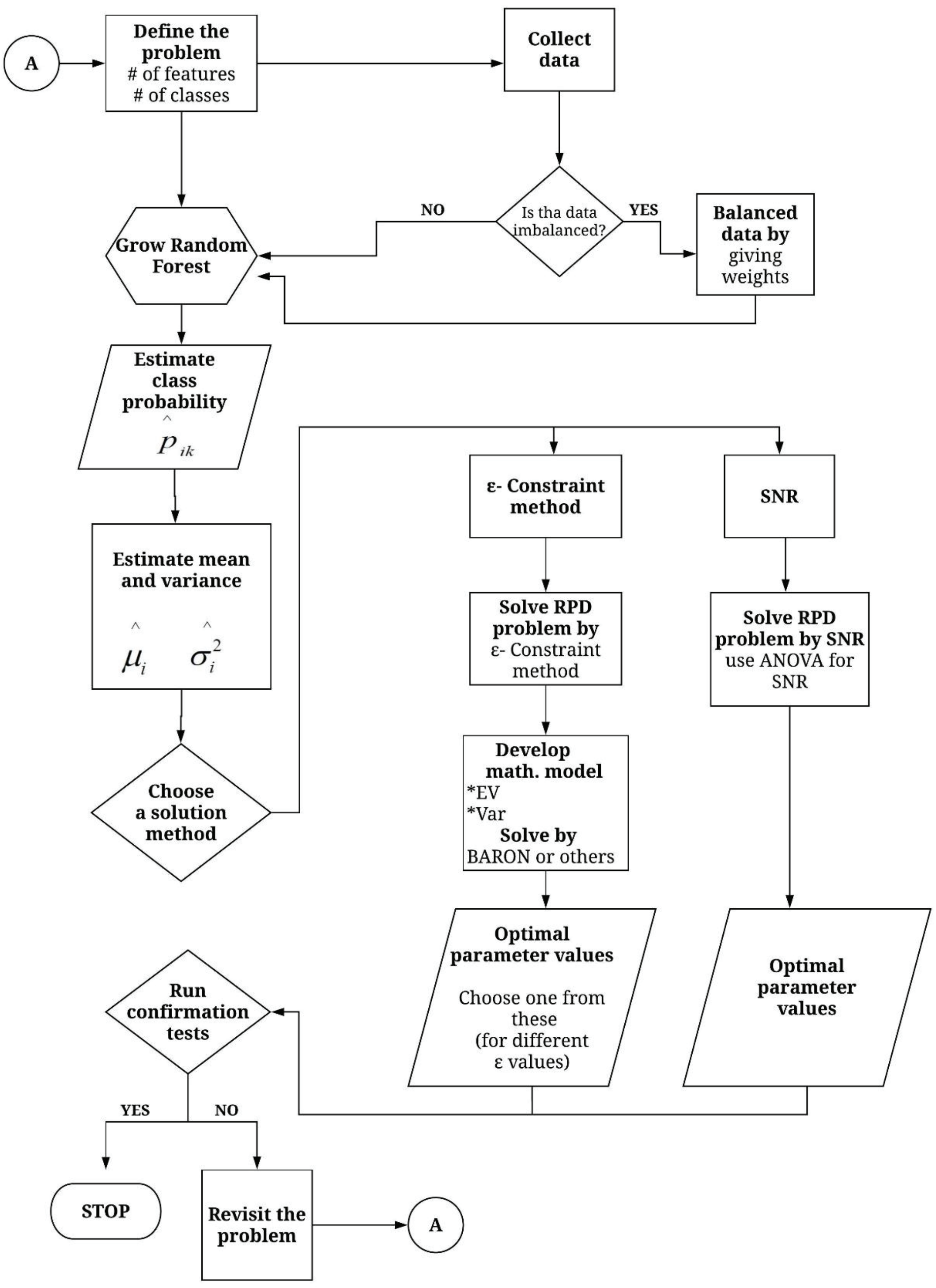


Figure 3.1. Flowchart of the Proposed Algorithm

3.1. Description of the Algorithm

Let x_1, x_2,…, x_p be controllable product or process design parameters (features, attributes or controllable factors). Let also Y be an ordinal response variable that takes values in K different categories.

Data sets are prepared as in Table 3.1 to be used in the Random Forest algorithm.

The proposed method enables us to find robust levels (settings) of the design variables (parameters) for ordinal categorical responses.

The steps of the proposed method for finding a robust product or process parameter design are given in the following part.

Here x_{ij} is the setting of variable x_j in experiment *i*. y_{iK}'s are 0 if experiment *i* is not classified as class *K* and 1 if experiment *i* is classified as class *K*.

Table 3.1. A Visual Overview of the Experimental Data Set Prepared for Random Forest Modelling and Its Results

Obs. No	Exp. No	Repli-cation No	Design Parameters				Response of the Designs in K Categories				Estimated Probability of Obtaining a Response in a Category			
			x_1	x_2	$\cdots$	x_p	Y_1	Y_2	...	Y_K	$\hat{P}(Y_1)$	$\hat{P}(Y_2)$	$\cdots$	$\hat{P}(Y_K)$
1	1	1	x_{11}	x_{12}	$\cdots$	x_{1p}	y_{11}	y_{12}	$\cdots$	y_{1K}	$\hat{p}_{11}$	$\hat{p}_{12}$	$\cdots$	$\hat{p}_{1K}$
2	1	2	x_{11}	x_{12}	$\cdots$	x_{1p}	y_{21}	y_{22}	$\cdots$	y_{2K}				
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$				
r	1	r	x_{11}	x_{12}	$\cdots$	x_{1p}	y_{r1}	y_{r2}	$\cdots$	y_{rK}				
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$(N\text{-}1)r+1$	N	1	x_{N1}	x_{N2}	$\cdots$	x_{Np}	$y_{((N-1)r+1)1}$	$y_{((N-1)r+1)2}$	$\cdots$	$y_{((N-1)r+1)p}$	$\hat{p}_{N1}$	$\hat{p}_{N2}$	$\cdots$	$\hat{p}_{Np}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$				
$N \times r$	N	r	x_{N1}	x_{N2}	$\cdots$	x_{Np}	$y_{(N\times r)1}$	$y_{(N\times r)2}$	$\cdots$	$y_{(N\times r)p}$				

Step 1: Collecting data and overcoming problems caused by missing values

Designing experiments and collecting data from these experiments is the first step of the RPD studies. In this phase, the aim of the experiments, inputs and outputs of the experiments, and how the data will be analyzed are determined (Montgomery, 2009). Montgomery lists the steps of this process as follows:

1. Identifying the problem
2. Determining the controllable and uncontrollable factors and their levels
3. Choosing the response variable
4. Selection of the experimental design
5. Performing the experiment and recording the data

Data analysis and recommendations follow these steps (Montgomery, 2013).

Firstly, the statement of the problem must be expressed clearly. Then, after determining the response to be reached and factors that affect these responses, the type of the experimental design is chosen. There are many experimental design types in literature, such as Factorial Design, Central Composite Design, and Optimal Design and so on. Afterwards, experiments are run based on the chosen design method and the results are recorded.

In our study, firstly, data that has p different product or process design parameters, one single response and N observations are collected according to a proper statistical experimental design. After obtaining the data set, it is examined and checked for whether there is missing data or not. The data might have some missing values in some or all of the variables. Fortunately, Random Forest can overcome the missing value problem easily. As mentioned in Section 2.2.1.5, there are three different ways to handle the missing value problem for Random Forest:

1. Imputing with column medians for continuous values
2. Imputing with the most frequent value for categorical values
3. Imputing by using proximities for both continuous and categorical values (more advanced)

The first two methods are faster than the third method. Breiman (2003) suggests that if missing values amount to less than 20% of the size of the data set, the best approach is to use the appropriate one of the first two methods. The third method works relatively slower, and needs up to 6 iterations, but it is more reliable. If there are more than 20% missing value, using a proximity-based approach may prove useful for getting more accurate results. The third method can be applied only for Random Forest, since it uses proximity matrix generated by the Random Forest method directly, if a software is used for that. It should be noted that when this method is used, the number of iterations needed to be repeated must be specified in the software used. Although the number of iterations varies depending on the used data set, Breiman (2003) says that, generally, the missing values could be filled in at most 6 replicates.

Step 2: Growing random forest trees and handling imbalanced data

Breiman (2001) explains the process of growing Random Forest trees as follows: The simplest random forest with random features is formed by selecting at a random small group of input variables at each node to split on. One of the most important points here is, while growing, trees are not pruned, and i.e. these are grown to their maximal size.

In our data sets, there are N different parameter settings, and each of these settings, $\mathbf{x}_i$, contains p different controllable product or process design parameters, $\mathbf{x}_i =(x_{i1}, x_{i2},\ldots,x_{ip})$, $i=1,\ldots,N$. When these N different $\mathbf{x}_i$ settings are repeated r times, they result in K number of different Y responses, $\mathbf{y}_i= (Y_{i1},\ldots,Y_{iK})$, $i=1,\ldots,N$. After all replications are finished, the size of our data set would be $\mathrm{N} \times \mathrm{r}$.

Firstly, a bootstrap is created with a size of N, which is equal to the size of the original data set by using only 63% of the data set with replacement. Secondly, as explained in Section 2.2.1, the numbers of split values are determined. Generally, the square root of the number of factors is selected as the attribute number for splitting. These attributes are selected in a completely random fashion. Among these randomly

selected attributes, the feature that gives the best split is chosen for the considered splitting (this procedure is repeated for each split).
According to this created subsample, a decision tree is grown to its maximum size, without pruning, using any tree methodology (CART, C4.5, C5.0, etc. in our study, CART methodology is used). In this manner, subsamples with a size of N are created as much as the number of trees that is desired to be in the forest, and trees are grown according to these subsamples, and, of course, they are independent of each other. This subsample randomization scheme is repeated with bagging to resample, with replacement, for each time a new individual tree is grown. The data whose class is to be decided is passed through each of these trees. Then finally, the predicted class for the data is the most frequent class predicted by these trees. In other words, the class which has the majority vote is determined as the class of the data.

Sometimes, the classes may not be distributed on a balanced basis in the data, i.e., some classes may recur many times, while others might be observed very rarely. In such a situation, to eliminate imbalancing, weights can be given to the class while the random forest is grown. So, if we have a less observable class in the data, especially if this class being of our interest, we can increase the cost of misclassification of this class. In this manner, Random Forest enforces the correct classification of the class. In addition to that, Random Forest does it automatically, so there is no need for any additional processing.

Step 3: Estimating probability of each class, and then calculating expected value and variance of each experiment

Estimating the probability of an experimental result is carried out by voting in Random Forest. Let $\hat{p}_{ik} = \mathrm{P}\left(Y = k \middle| \mathbf{x}_i = \left(x_{i1}, x_{i2}, \ldots, x_{ip}\right)'\right)$ be the proportion of trees that predict the class of experiment $\mathbf{x}_i$ as k, when the data is passed through the trees in the forest. It should be noted that this probability is calculated only in trees where $\mathbf{x}_i$ is out-of-bag (Li, 2013).

$$\hat{p}_{ik} = \frac{\text{Number of trees that estimate the class of the experiment } i \text{ as } k}{\text{Number of trees in the forest}} \tag{3.1}$$

where $0 \le \hat{p}_{ik} \le 1$, $k = 1,\ldots,K.,\ i = 1,\ldots,N$

This estimated probability is also used for the calculation of expected value and variance for each factor combination. By using the estimated probability of experimental results for each class, we calculate the expected class and the variance for each observation in the data set, by using the Equations (3.2) and (3.3).

$$\hat{\mu}_i = \hat{E}(Y_i) = \sum_{k=1}^{K} k\,\hat{\mathrm{P}}\left(Y = k \middle| \mathbf{x}_i\right) \tag{3.2}$$

$$\hat{\sigma}_i^2 = \hat{V}(Y_i) = \hat{E}\left(Y_i^2\right) - \left[\hat{E}(Y_i)\right]^2 = \left[\sum_{k=1}^{K} k^2\,\hat{\mathrm{P}}\left(Y = k \middle| \mathbf{x}_i\right)\right] - \hat{\mu}_i^2 \tag{3.3}$$

where

i	= Number of experiment (i = 1,2,…,N)
k	= Class number (k =1,2,…,K)
$\hat{\mu}_i$	= Estimator of the mean class for experiment i
$\hat{\sigma}_i^2$	= Estimator of the variance of the class for experiment i
$\hat{\mathrm{P}}\left(Y = k \middle| \mathbf{x}_i\right)$	= Estimator of the probability of result of experiment i being k

Step 4: Find the optimal product/process design parameter settings for the desired mean and minimum variance

For the purpose of obtaining settings of the controllable factors that ensure desired expected value and minimum variance, two different methods can be used. These are finding the optimal values of parameters by (1) non-linear multi-objective optimization (ε-Constraint method) for both $\hat{E}(Y_i)$ and $\hat{V}(Y_i)$, and (2) maximization of SNRs. In the ε-Constraint method, the model is analyzed according to the expected value and the variance, separately, while these two are combined into a single performance criterion, SNR, in maximization SNR method.

a) Obtain the optimal parameter settings by using ε-Constraint method

The first method uses the ε-Constraint method to obtain the desired expected value (one objective) and minimum variance (the other objective) for all possible values of the design parameters. We suggest use of the ε-Constraint Method as explained in Section 2.1.2.1 to solve the problem. We use the method in our study as follows: At first, we try to obtain class probabilities for each experimental design point from the Random Forest, and by using these probabilities, we calculate the expected value and the variance corresponding to each of these design points. After that, we fit regression models for the expected value and the variance. We obtain the mathematical models for the expected value and the variance using least squares regression. These models include both main and interaction effects of the factors. Then, a mathematical model is set, so that the variance is minimized as the objective function, and the expected value is set as a constraint in the form of a difference from the desired target value being less than or equal to ε. By using these models, the ε-Constraint method is tried to be solved with an appropriate algorithm and software (in our study, we use MATLAB/BARON).

As a result, we get the parameter values that minimize the variance and also simultaneously provide the expected values in the epsilon neighborhood of the target value (the best case is obtained when epsilon is equal to 0).

The optimization models for different problem types are as given Table 3.2.

Table 3.2. Optimization models for the different problem types

Problem Types	**Mathematical Model**
Smaller-the-Better	$Min\ \hat{E}(Y) = f(x_1, x_2, \ldots, x_p)$ $Min\ \hat{V}(Y) = g(x_1, x_2, \ldots, x_p)$ $s.t.$ $l_i \le x_i \le u_i\ ,\quad i = 1,2,\ldots,p$ $\hat{E}(Y) \ge 0$ $\hat{V}(Y) \ge 0$
Larger-the-Better	$Max\ \hat{E}(Y) = f(x_1, x_2, \ldots, x_p)$ $Min\ \hat{V}(Y) = g(x_1, x_2, \ldots, x_p)$ $s.t.$ $l_i \le x_i \le u_i\ ,\quad i = 1,2,\ldots,p$ $\hat{E}(Y) \ge 0$ $\hat{V}(Y) \ge 0$

Here, l_i and u_i are lower and upper limits of x_i, respectively.

For both types of the problem, the model with the ε-Constraint Method is as follows:

$$\begin{aligned}
&Min\ \hat{V}(Y) = g(x_1, x_2, \ldots, x_p) \\
&s.t. \\
&\left|\hat{E}(Y) - t\right| \le \varepsilon, \quad \hat{E}(Y) = f(x_1, x_2, \ldots, x_p) \\
&l_i \le x_i \le u_i\ ,\quad i = 1,2,\ldots,p \\
&\hat{E}(Y) \ge 0 \\
&\hat{V}(Y) \ge 0
\end{aligned} \tag{3.4}$$

where t is the target value for the expected value, and ε is the upper bound for the difference between target value and estimated expected value.

The general form of the ε-Constraint formulation of the problem is as given in Model (3.4). But, to achieve more robust solutions, we also add the difference between the expected value and the target value to the objective function after multiplying it with a very small coefficient a (to avoid favoring $\hat{V}(Y)$ in the tradeoff between the variance and the expected value). So, the new formulation of the problem for the ε-Constraint Method is as given in Model (3.5).

$$\begin{aligned}
&Min\left\{\left[\hat{V}(Y)=g\left(x_1,x_2,...,x_p\right)\right]+\left[\left|\hat{E}(Y_i)-t\right|\times a\right]\right\}\\
&s.t.\\
&\left|\hat{E}(Y)-t\right|\leq \varepsilon \ , \quad \hat{E}(Y)=f\left(x_1,x_2,...,x_p\right)\\
&l_i\leq x_i\leq u_i \ , \quad i=1,2,...,p\\
&\hat{E}(Y)\geq 0\\
&\hat{V}(Y)\geq 0
\end{aligned} \qquad (3.5)$$

We solve the model as described in Section 2.1.2.1 and obtain different solutions based on different ε values. Then the most appropriate solution in the direction of our (or the decision makers) aims is chosen.

b) Obtain the optimal parameter settings by using SNR

In an alternative method, Taguchi's SNR values are calculated by using $\hat{E}(Y_i)$ and $\hat{V}(Y_i)$, according to the problem type (smaller-the-better, larger-the-better). The SNR formulas are given in Section 2.1.2.2 in Table 2.2.

An ANOVA or regression model is fit to these calculated SNR values. For this purpose, one can use stepwise regression. The stepwise regression adds the most significant factors and significant interactions, if any, to the model, and removes relatively insignificant factors from the model. This model can be taken as a good start in finding the final model based on adjusted R-squared and predicted R-squared measures. After obtaining a satisfactory starting point by using stepwise regression, we try to achieve more powerful models by adding or removing factors and their interactions from the model. Finally, with Minitab's response optimizer (Minitab 18 Statistical Software, 2018) or another appropriate algorithm, the best parameter setting can be found by maximization of SNR (both for smaller-the-better and larger-the-better). Also, in this step, for the response optimizer, different starting points are tried in order to achieve the best parameter settings.

Step 5: Confirmation of results and revisiting the problem

In confirmation step, the decision maker has to perform replicated experiments at the optimal settings of the design parameters to make sure the solution works as predicted. To achieve adequate power for the confirmation test, we want enough replications but at the same time, we do not want too many replications to avoid wasting time. Thus, the number of replications and power of the confirmation test are important decisions for this step. For that, we can apply an appropriate multinomial test as such the chi-square (χ^2) test for Goodness-of-Fit. The χ^2 test uses sample data sets to test hypothesis about the model used (Gravetter and Wallnau, 2014). To find the confirmation intervals corresponding to any given number of replications while applying χ^2 test, firstly we test the design parameters which correspond optimal solutions obtained from the algorithms for *n* times. Then we define the observed probability of each class *k* from those replications as $\mathbf{p}_i=(p_{i1},\ldots,p_{ik})$, $i=1,\ldots,n,$ and $\sum_{j=1}^{k} p_{ij} = 1$ for each replication (Read and Cressie, 1998). After we obtain the observed probabilities, Read and Cressie (1988) suggest that a null hypothesis is defined for χ^2 test as H$_0$: $\mathbf{p}_i = \hat{\mathbf{p}}_i$, where the vector $\hat{\mathbf{p}}_i$ is the estimated probabilities obtained from the algorithm. Gravetter and Wallnau (2014) define this null hypothesis as there is no difference between the observed probabilities and the estimated probabilities. It is obvious that the hypothesis probabilities are not expected to be exactly equal to estimated probabilities. Thus, if there are small discrepancies between $\hat{\mathbf{p}}_i$ and $\mathbf{p}_i$ we obtain a small value for χ^2 and we conclude that the algorithm works properly for this data set (fail to reject H_0). On the other hand, when there are large discrepancies between $\hat{\mathbf{p}}_i$ and $\mathbf{p}_i$, H_0 is rejected and we conclude that algorithm does not work properly for this data set. According to Gravetter and Wallnau (2014), to obtain particular value of χ^2, we should refer to a chi-square distribution, χ^2, which is the most well-known goodness-of-fit statistics, introduced by Pearson at 1900, and it is calculated as:

$$\chi^2 = n \times \sum_{i=1}^{k} \frac{\left(\mathbf{p}_i - \hat{\mathbf{p}}_i\right)^2}{\hat{\mathbf{p}}_i} \tag{3.6}$$

Firstly, the decision maker decides on the desired confirmation significance levels and specify the critical value corresponding to these levels via chi-square distribution. If the result obtained from Equation 3.6 is less than or equal to this critical value, it is concluded we fail to reject H_0. If this is not the case, H_0 is rejected and it means that there is a large difference between the observed and estimated probabilities. Or, to decide on the number of replications, *n*, firstly we decide on the confirmation significance level to fail to reject H_0 and then find the critical value corresponding these levels via chi-square distribution. According to this critical value, the number of replications, *n*, is calculated based on Equation 3.7.

$$n = \frac{\chi^2}{\sum_{i=1}^{k} \frac{\left(\mathbf{p}_i - \hat{\mathbf{p}}_i\right)^2}{\hat{\mathbf{p}}_i}} \tag{3.7}$$

In the case where expected probability (or expected count) of at least one class is 0, Chi-Square test cannot be usable. For such cases, Exact Multinomial Test for Goodness-Of-Fit Test for Discrete Multivariate Data can be usable.

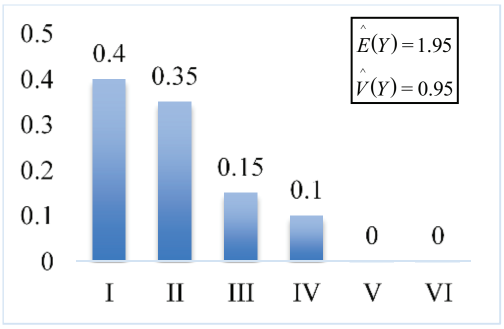
We can offer some suggestions for the case where the null hypothesis is rejected.

1. Firstly, it is checked whether the steps of the algorithm are applied correctly or not. If there is no mistake in the steps, new data can be collected not only corresponding to the optimal solution but also at different parameter designs, if it is possible. New parameter settings can be chosen so as to have an orthogonal experimental design. After collecting the new data, the algorithm is reapplied on the cumulative data.
2. The regression models built for EV, Var and SNR should be controlled for R-sq, R-sq(adj), and R-sq(pred) values and lack-of-fit.

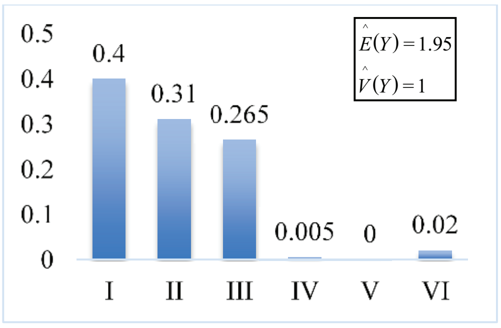
    - If the models exhibit lack-of fit, some interactions and quadratic terms can be added to the model (Minitab, 2018).
    - If there is no lack-of-fit, but R-sq, R-sq(adj), and R-sq(pred) values are less than the desired ones, one can consider adding one or more new parameters to the regression model, if it is possible (Montgomery, 2013). This, of course may require to do a new set of experiments with the new parameters added.
    - If there is no lack-of-fit, and R-sq, R-sq(adj), and R-sq(pred) are at satisfactory levels, the regression models can be used in solving the problem.
3. At the optimization step, different starting points can be tried.

After these suggestions and alternative ways are tried, the algorithm is rerun and confirmation tests are performed. If we can obtain desired confirmation values after rerunning the algorithm, we can say that the algorithm works properly for this data set. On the other hand, if we still do not reach the desired confirmation values, we can conclude that the algorithm does not work properly for the data set.

In this study, the results of our proposed model are compared with themselves; namely, RF_E, RF_L, RF_P, and RF_X (these stand for different weighting approaches), and with Karabulut (2013) results in general. As the comparison criteria of the optimal results, the expected value and variance are also used in addition to the estimated probabilities. Because the probability of observing class is not a sufficient measure, which can be misleading. In some cases, while the estimated probability is high, the variance might be high as well.



b) Situation I



a) Situation II

Figure 3.2. Exemplary situations where the probabilities are the same, but the variance is different

For example, for the smaller-the-better type problem depicted in the Figure 3.2, the two situations are shown with equal probabilities of Class I (desired class) and also their expected values are the same. But in the second situation, the variance is higher. This is not the preferred case; that is why, to find the best parameter settings, we need to consider both the mean and the variance of the results.

To find the estimated probability, expected value and variance for the optimal parameter settings, three different methods can be used:

1. If the optimal parameter settings are previously tried at the initial experiment, the probability of observing a class can be estimated by dividing the number of observations at that class by the total number of observations at the optimal parameter settings. After that, by using these estimated probabilities for all classes, the expected value and the variance can be estimated using Equations (3.2) and (3.3).

2. At first, the class probabilities, expected values and variances are estimated for all experimental points. Then, regression models are fit for the expected value and the variance. Using these regression models, expected value and variance at the optimal parameter settings can be estimated.
3. Estimated probabilities at the optimal parameter settings are obtained from the Random Forest results, and the expected values and the variances are estimated by using Equation (3.2) and (3.3).

In this study, we use the second and third method, since the optimal solution may not always be among the tested cases, and the probability of the untried experiment cannot be calculated with the first method. Since the data that we examine in this study are relatively small, using the second and third method can be more appropriate. If the data were big, we would have calculated the probability (as well as the expected value and the variance) by using only the data itself, as in Method 1.

Now, to test our method, we apply it on three different types of cases. Two of them are of the larger-the-better type and the other one is of the smaller-the-better type. These cases are also used by Karabulut (2013) to compare different methods, namely LRMO, AA, WSNR, SS, and WPSS (see Section 2.1.1). We also want to compare our results to those of these different methods to understand the performance of our method.

3.2 Illustrative Case Study I: Surface Defects Case

In this section, our proposed method is applied to an example data set, which has been used by Phadke (1989) and Karabulut (2013) for RPD of polysilicon deposition process, and comparing four different RPD methods, respectively.

Step 1: Collecting data and overcoming problems caused by missing values

The data set presented below has been collected by Phadke (1989) to improve the polysilicon deposition process. For that purpose, he uses Taguchi Robust Design in his study. In this data set, there are six controllable factors: Deposition temperature (A), deposition pressure (B), nitrogen flow (C), silane flow (D), setting time (E), and cleaning method (F). The level of surface defects on unit area is set as the quality characteristic, i.e. the amount of surface defects on the designated area. All controllable factors have three different levels (1, 2 and 3), and the quality characteristic is categorized as five different, ordered levels. Briefly, the values of all factors are defined by Phadke (1989) as in Table 3.3.

Table 3.3. Controllable Factors and Their Levels

Factors	**Levels**		
	1	2	3
A. Deposition temperature (^oC)	T_0-25	T_0	T_0 +25
B. Deposition presure (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. Setting time (min)	t_0	t_0+8	t_0+16
F. Cleaning methods	None	CM_2	CM_3

Before starting the study, the order of levels of factor C is changed, since the original levels of it were not in any order. The new levels of factor C are reorganized by Karabulut (2013) as in Table 3.4, and factor C will be indicated as C' with its new level order:

Table 3.4. Reorganized Levels of Factor C

Levels	**C**	**C'**
N_0	1	3
N_0-150	2	1
N_0-75	3	2

When the quality characteristic of the product is tested according to these factors at their different levels, the numbers of surface defects on the different areas of the product are counted, and according to these counted numbers, responses are divided into five different classes by Phadke (1989) as shown in Table 3.5.

Table 3.5. Number of Surface 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

In this data set, the aim is to reach the wafers which have the minimum number of surface defects, so it is a smaller-the-better type problem. Responses of the data are created in a sequential order, Class I refers to the smallest number of surface defects, and Class V refers to the highest number of surface defects. So, Class I is the preferred class. In other words, while class labels decrease, quality increases.

18 experiments were carried out with factor levels depending on the L_{18} orthogonal array, and each experiment runs for 9 times. Finally, responses of these 9 runs for each experiment are classified according to the number of surface defects. The number of observations for each class for the eighteen experiments are tabulated in Table 3.6.

Table 3.6. Experimental Design and Data Collected for the Surface Defects Case

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

Since there is no missing value in the data, no manipulation is done for that purpose.

Step 2: Growing random forest trees and handling imbalanced data

The data collection experiment is designed so as to run 9 times for 18 different factor level combinations, so we have 162 observations in our data set. This data set is analyzed with the Random Forest algorithm by using R-studio software (R Core Team, 2017).

First of all, Random Forest creates random subsamples each with a size of 162, which is equal to the size of the original data set, with replacement. Secondly, the numbers of attributes to use at each split are determined. As mentioned before in Section 2.2.1, this number is calculated using $\lfloor \sqrt{p} \rfloor$, where p is the number of attributes, and for this problem, the number is $\lfloor \sqrt{6} \rfloor$=2. Also, in R, the best split value is determined with respect to OOB error estimate, and the result is 2 for this value as well. The best split value graph in R is shown in Figure 3.3.

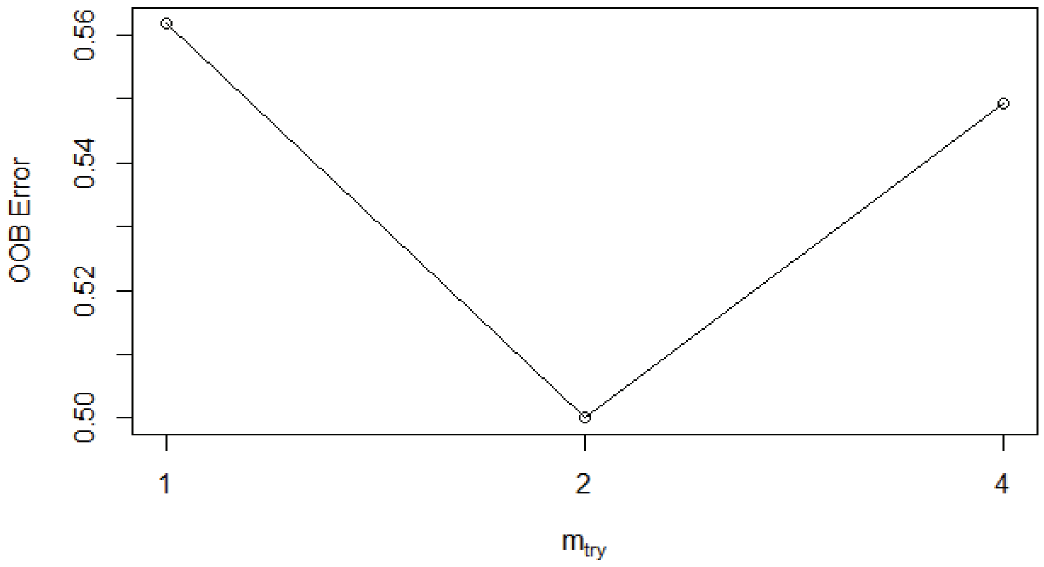


Figure 3.3. The Best Split Value versus OOB Error Rate for the Surface Defect Case

Moreover, the best number of trees in the forest is tested with different values between 100-10000, and the best value is set as 700 according to both OOB error rate and consuming of time.

For this data set distribution of trials among classes is shown in Figure 3.4. As seen in the figure, Class I, our class of interest, has the maximum number of observations, 49, 30% of the data set. The other classes constitute 17%, 17%, 14% and 22% of the data set, respectively. Since all classes do not have the same number of trials, we can say that this data set is imbalanced. By giving weights to the classes, Random Forest overcomes this situation; that is, imbalancing. To observe the effect of giving weights to the classes, we are going to try Random Forest both with unweighted classes and with those having different weights. For that, firstly we apply regular Random Forest to the data by giving equal weights to the classes, and then we give more misclassification cost to our class of interest, Class I, than the other classes with different ratios. Then, we observe the effect of the weight on responses.

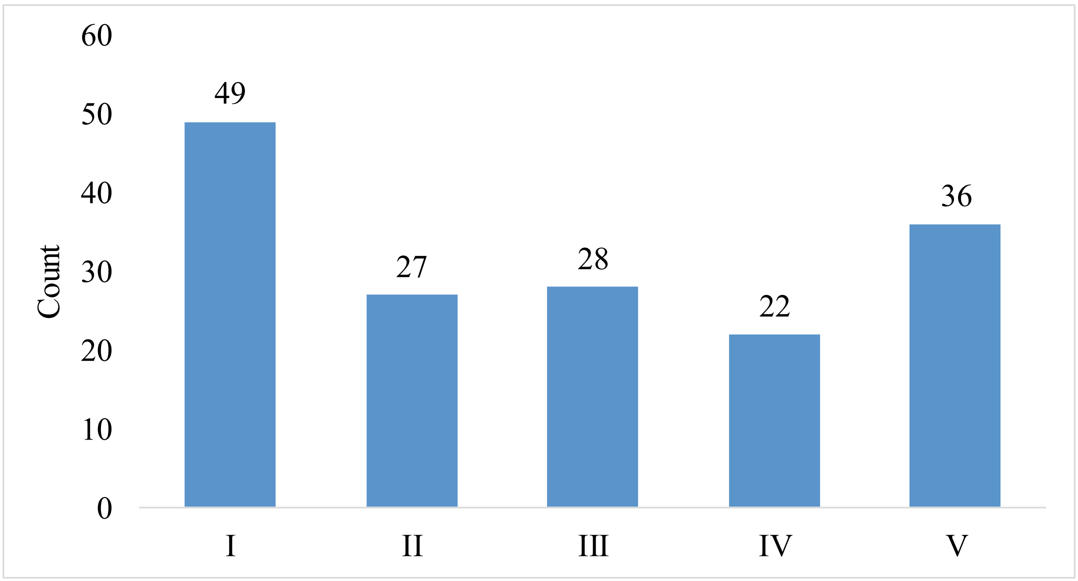


Figure 3.4. Distribution of the Data to the Classes for the Surface Defects Case

Giving the weights to the classes is an important mean provided by Random Forest to handle imbalanced data. Determination of the weights is subject to the users. We try different weighting approaches to study their effects on the results.

We first give equal weights to the classes, and apply regular Random Forest. Secondly, we give linearly distributed weights to the classes in the increasing order of class importance. For this example, since it is a Smaller-the-Better type problem, weights of classes change according to a linearly decreasing order. Thirdly, we give fifty percent of the total weights to our class of interest, and then, the rest of the weights is distributed to the remaining classes equally (this distribution strategy performs a piecewise linear function). Lastly, we will give exponential weights which are proportional to the order of the classes (for this problem, as it is Smaller-the-Better type problem, weights are changed according to an exponentially decreasing order).

The weights created according to these strategies are given in Table 3.7, as well as their abbreviations.

Table 3.7. Class Weights of the Random Forest Applications for the Surface Defects Case

Weighting Strategies	**RF Abbreviation**	**Weights of Classes**				
		I	**II**	**III**	**IV**	**V**
Equal Weights	RF_E	0.20	0.20	0.20	0.20	0.20
Linear Weights	RF_L	0.33	0.27	0.20	0.13	0.07
Piecewise Linear Weights	RF_P	0.500	0.125	0.125	0.125	0.125
Exponential Weights	RF_X	0.64	0.23	0.09	0.03	0.01

According to these weights, OOB error graphs of Random Forests formed with 700 trees and the split value equal to 2 are as given in Figures 3.5-3.8.

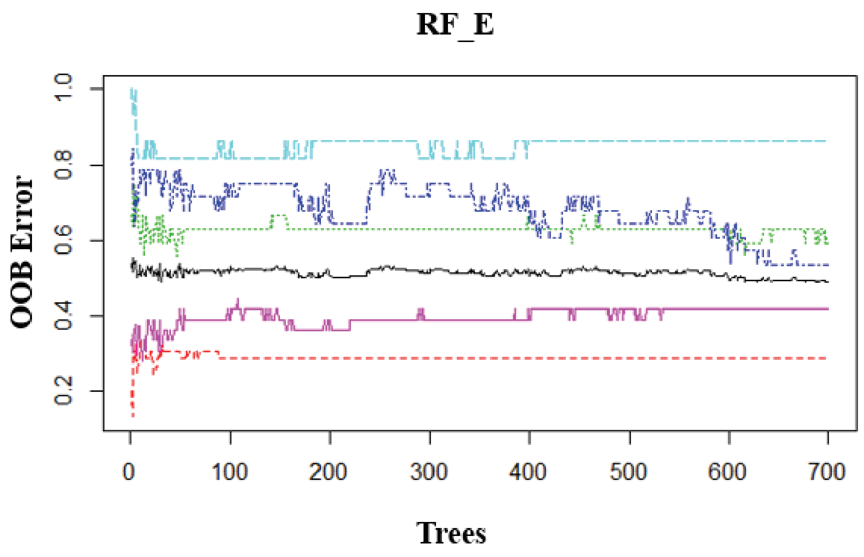


Figure 3.5. OOB Error Graph of RF_E for the Surface Defects Case

According to this graph, the black line is the OOB error estimate line of RF_E with 48.77% on the average. The red line signs Class I with an average of 29% OOB error rate (Class I has the minimum OOB error rate among other classes). The pink line refers to Class V, and it has 42% OOB error rate on the average. The blue and green lines refer to Class III and Class II with averages of 53% and 59% OOB error rates, respectively. Lastly, the cyan line belongs to Class IV, and its average OOB error rate is the maximum; that is, 86%. So, we can see that this Random Forest can make a satisfactory prediction for Class I, but might not do so for Class IV. For the other class, prediction accuracy would be neither good nor bad.

Now, we look at the weighted Random Forest results. First, we examine the OOB graph of RF_L.

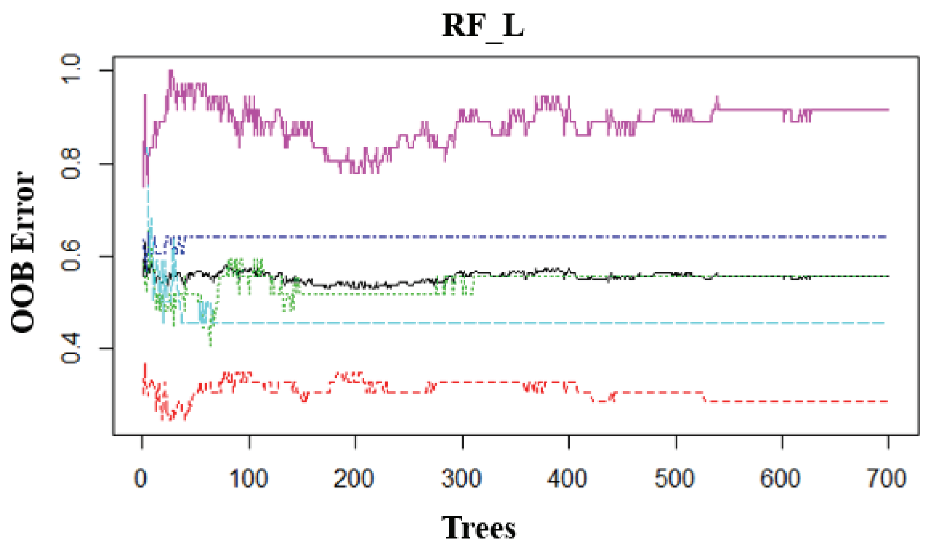


Figure 3.6. OOB Error Rate Graph of RF_L for the Surface Defects Case

Here, we see that OOB error estimate of the Random Forests have increased compared to those of RF_E. The average OOB error estimate of the Random Forest has changed from 48.15% to 53.09% (black line). And the average OOB error rate of Class I does not change, it is still 29% (red line). The average OOB error rates of Class II and Class IV have decreased (green and cyan lines, respectively), with the OOB error rate of Class II changed from 59% to 55%, and that of Class IV changed from 86% and 45%. The OOB error rates of Class III and Class V, on the other hand, have increased (blue and pink lines, respectively). Their average OOB error rates have increased from 53% to 64% for Class III, and from 42% to 92% for Class V. So, we conclude that giving linearly decreasing weights does not affect the prediction accuracy of Class I, but affect those of the other classes.

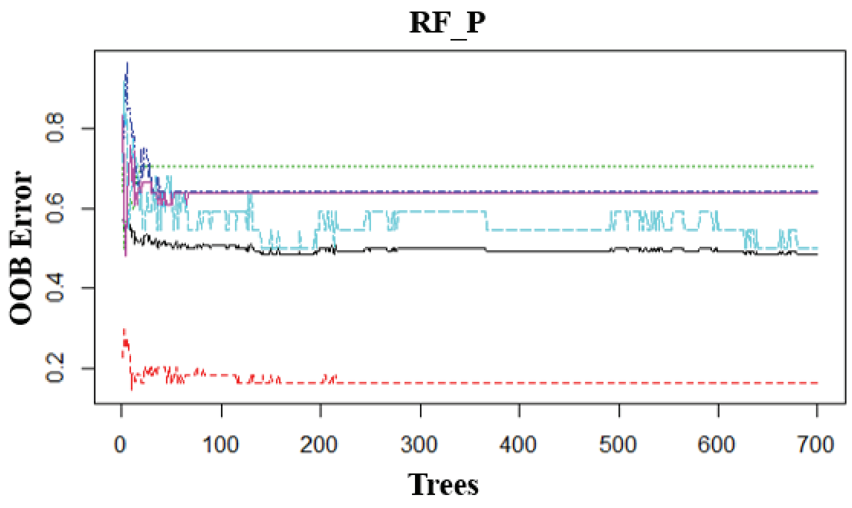


Figure 3.7. OOB Error Rate Graph of RF_P for the Surface Defects Case

When the Figure 3.7 is examined, it is seen that the OOB error estimate of RF_P (black line – 48.77%) is equal to the OOB error estimate of RF_E, and better than the OOB error estimate of RF_L. And also another good news is that the OOB error rate of our class of interest, Class I, is very low (red line - 16%). The average OOB error rate of Class V has also decreased compared to RF_L (pink line – 63%). The OOB error rate is fixed for Class III (blue line – 64%). While the average OOB error rate of Class IV has decreased compared to that of RF_E and it has increased when compared to that of RF_L (cyan line – 50%), the average OOB error rate of Class II has increased compared to both RF_E and RF_L (green line – 70%). So, we can say that, this Random Forest can make better predictions than RF_E and RF_L for Class I; that is, our class of interest.

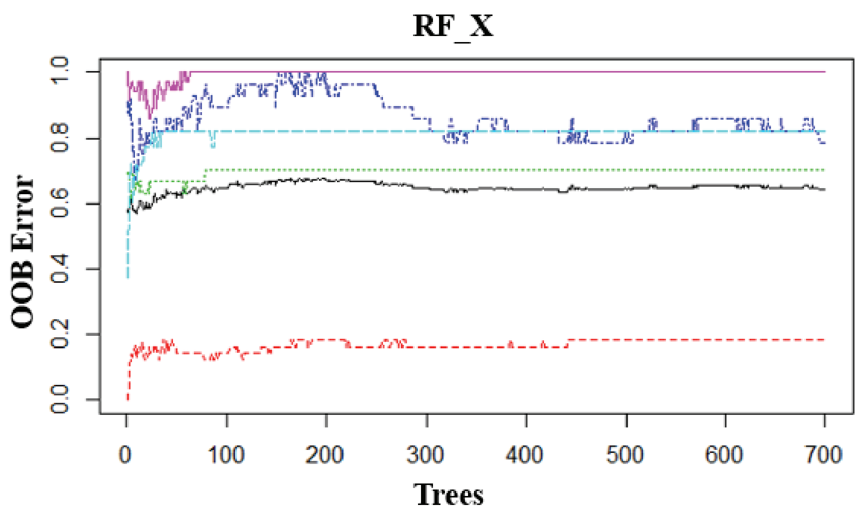


Figure 3.8. OOB Error Rate Graph of RF_X for the Surface Defects Case

Lastly, we analyze the Random Forest results with exponential weights, RF_X. As we expect, the OOB error rate of Class I is relatively low, but those of the other classes are high, because we give sixty-four percent of the weight to Class I, and the remaining weights are distributed again exponentially among classes according to the class order. So, the red line signs the Class I, and its OOB error rate is 18% on the average. Actually, this value is higher than the value of RF_P. And also, here, OOB error estimate of the Random Forest is the maximum, 61.73% (black line). The green, blue, cyan and pink lines show Class II, Class III, Class IV and Class V, respectively, and their average OOB error rates are 70%, 79%, 82% and 100%, respectively.

In short, when we evaluate the OOB graphs of these four different weighted Random Forests, we observe that both RF_E and RF_P have the minimum OOB error estimate (48.77%). Also, RF_P is the Random Forest whose OOB error rate of Class I is the minimum. However, the OOB error rate of Class II, which is the class closest to our class of interest, is the maximum in RF_P. In RF_X, the OOB error rate of Class I is better than those of RF_E and RF_L, but worse than that of RF_P, and the OOB error rate of the other classes are quite high. In addition to that, RF_X is the Random Forest that has the maximum OOB error estimate (64.2%). In RF_E and RF_L, the OOB error estimate of Random Forests, and the OOB error rate of classes including Class I are average. The type of weight we will choose varies according to our aim. We can select RF_P, if the performance of Random Forest and prediction performance of our class of interest are important. However, if the prediction performance of other classes is also essential for us, we can also choose RF_E. If we cannot decide at this stage, we have to proceed to the optimization step.

Additionally, we can analyze classification performance of these Random Forests by using the most well-known performance measures of classification power (Ferri, 2009) mentioned in Section 2.2.1.6. Also, we can compare Random Forest and Logistic Regression as classification methods by using these performance measures. Comparison of Random Forest and Logistic Regression based on some common measures is as presented in Table 3.13. Let us give, firstly, confusion matrices of both methods. For Random Forests, we give four different confusion matrices in Table 3.8 – 3.11 according to their weighted schemes.

Table 3.8. Confusion Matrix of RF_E for the Surface Defects Case

		Predicted					
	Classes	I	II	III	IV	V	Sum
Actual	I	35	3	6	0	5	49
	II	5	12	6	2	2	27
	III	4	3	17	1	3	28
	IV	1	0	3	8	10	22
	V	0	0	4	7	25	36
	Sum	45	18	36	18	45	162

Table 3.9. Confusion Matrix of RF_L for the Surface Defects Case

		Predicted					
	Classes	I	II	III	IV	V	Sum
Actual	I	40	7	2	0	0	49
	II	6	18	0	2	1	27
	III	6	10	10	1	1	28
	IV	4	0	3	12	3	22
	V	7	1	3	12	13	36
	Sum	63	36	18	27	18	162

Table 3.10. Confusion Matrix of RF_P for the Surface Defects Case

		Predicted					
	Classes	I	II	III	IV	V	Sum
Actual	I	47	0	2	0	0	49
	II	16	8	0	2	1	27
	III	15	1	10	1	1	28
	IV	4	0	3	12	3	22
	V	8	0	3	12	13	36
	Sum	90	9	18	27	18	162

Table 3.11. Confusion Matrix of RF_X for the Surface Defects Case

		Predicted					
	Classes	I	II	III	IV	V	Sum
Actual	I	48	0	1	0	0	49
	II	16	11	0	0	0	27
	III	19	3	6	0	0	28
	IV	5	9	2	6	0	22
	V	11	13	0	12	0	36
	Sum	99	36	9	18	0	162

The data can also be analyzed using by Logistic Regression as shown by Karabulut (2013). Table 3.12 show confusion matrix of the Logistic Regression results.

Table 3.12. Confusion Matrix of Logistic Regression (LR) for the Surface Defects Case

		Predicted					
	Classes	I	II	III	IV	V	Sum
Actual	I	35	0	10	0	4	49
	II	11	0	14	0	2	27
	III	5	0	15	0	8	28
	IV	1	0	8	0	13	22
	V	0	0	9	0	27	36
	Sum	52	0	56	0	54	162

Confusion matrices are constructed based on testing all of the data set, not the test set. Since we use all of the data for learning, do not split them into train and test sets (because Random Forest does not need splitting. Random Forest divides the data as train and test by itself).

Measures calculated based on the confusion matrices in Table 3.7 - 3.12 are tabulated as in Table 3.13.

As it is seen from Table 3.13, the accuracies of all weighted schemes except for RF_X are better than the accuracy of Logistic Regression. Since our data is not exactly balanced, we use balanced accuracy, and the balanced accuracies of all Random Forests are greater than the balanced accuracy of Logistic Regression. Similarly, accuracy-based kappa values of Random Forests, except for RF_X, are better than Logistic Regression's Kappa. As we expected, Kappa values, which are calculated with the balanced accuracies, are better than Logistic Regression for all Random Forests. F-Score of Logistic Regression is better than RF_X, but worse than the other Random Forests. AUC is the measure that the Logistic Regression shows better performance than weighted Random Forests. When we look at all values, we can say that generally Random Forests perform better than Logistic Regression.

In addition to these comparisons, class based measures can also be compared. Class based comparison table is given in Table 3.14. Here, we see that, for our class of interest, Class I, the method that shows the best performance according to Recall, Precision and Specificity are RF_X, RF_E, and RF_E, respectively.

We understand that, by changing the weights, we can increase or decrease the performance of a Random Forest. Besides, Random Forest uses this property very easily and fast.

When we evaluate all these performance measures, the best performing method varies according to each performance measure. However, all Random Forests generally perform better than Logistic Regression. It may be misleading to choose the best method in this stage. But, these performance measures can also be used as an indicator in the optimization step, while deciding on the method to be selected.

When we look at the importance of variables (all variable importance charts are given in Appendix C1) we see that the order is almost the same for RF_E, RF_L, and RF_P, but differs for RF_X. Although there are some differences, the most important variable is A and the second important variable is B for all RFs and for both measures (Mean Decrease Accuracy and Mean Decrease Gini). Moreover, the least important attribute is generally D. We can conclude that in the case where some factors have to be chosen for the trials, these important factors can be selected first.

Table 3.13. Comparison of Random Forests and Logistic Regression According to Performance Measures for the Surface Defects Case

RFs	**Acc**	**Balanced Acc**	**Kappa**	**Balanced Kappa**	**F-Score**	**G-Mean**	**AUC**
RF_E	60%	71%	49%	62%	60%	70%	81%
RF_L	57%	62%	45%	51%	56%	68%	77%
RF_P	56%	67%	41%	57%	53%	63%	75%
RF_X	44%	75%	25%	67%	35%	0%	69%
LR	47%	50%	32%	35%	41%	0%	80%

Table 3.14. Comparison of Random Forests and Logistic Regression According to Class-Based Performance Measures for the Surface Defects Case

Classes	Recall					Precision					Specificity				
	RF_E	RF_L	RF_P	RF_X	LR	RF_E	RF_L	RF_P	RF_X	LR	RF_E	RF_L	RF_P	RF_X	LR
I	71%	82%	96%	98%	71%	78%	63%	52%	48%	67%	91%	80%	62%	55%	85%
II	44%	67%	30%	41%	0%	67%	50%	89%	31%	0%	96%	87%	99%	81%	100%
III	61%	36%	36%	21%	54%	47%	56%	56%	67%	27%	86%	94%	94%	98%	69%
IV	36%	55%	55%	27%	0%	44%	44%	44%	33%	0%	93%	89%	89%	91%	100%
V	69%	36%	36%	0%	75%	56%	72%	72%	0%	50%	84%	96%	96%	100%	79%

Step 3: Estimating the probability of each class, and calculating expected value and variance for each experiment

Probabilities that are obtained from RFs (RF_E, RF_L, RF_P, and RF_X) and expected value, variance and SNR values that are based on these probabilities are given in Appendix C.2. For calculating the SNR, Equation (2.5) is used (since the problem is smaller-the-better type).

For all Random Forest trees except RF_E, the first experiment, $A_1B_1C_3^{'}\left(C_1\right)D_1E_1F_1$, is the best design among the tested ones according to the probability of our class of interest, expected value, variance, and SNR. For RF_E, the tenth experimental design, $A_1B_1C_2^{'}\left(C_3\right)D_3E_2F_1$, is the best experiment among the tested ones. The tenth experiment is also the best design for RF_P and RF_X. Moreover, RF_P has another best design; it is the eleventh experiment, $A_1B_1C_3^{'}\left(C_1\right)D_1E_3F_2$. At this parameter designs, the probability of Class I is at its highest value, and the expected value is at the target value; that is, 1, for all weighted schemes except for RF_E. And also for these parameter designs, the variance is the minimum, and the SNR value is the maximum for all Random Forest applications. The best parameter designs among the ones tested for all Random Forest applications are given in Table 3.15.

Table 3.15. The Best Parameter Designs Obtained from Different Random Forest Applications for the Surface Defects Case

RFs	Factors							Probability Estimates					$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	A	B	C'	C	D	E	F	$\hat{P}(Y = I)$	$\hat{P}(Y = II)$	$\hat{P}(Y = III)$	$\hat{P}(Y = IV)$	$\hat{P}(Y = V)$			
RF_E	1	1	2	3	3	2	1	0.9971	0	0.0029	0	0	1.0057	0.0114	-0.0980
RF_L	1	1	3	1	1	1	1	1	0	0	0	0	1	0	0
RF_P	1	1	3	1	1	1	1	1	0	0	0	0	1	0	0
	1	1	2	3	3	2	1	1	0	0	0	0	1	0	0
	1	2	3	1	1	3	2	1	0	0	0	0	1	0	0
RF_X	1	1	3	1	1	1	1	1	0	0	0	0	1	0	0
	1	1	2	3	3	2	1	1	0	0	0	0	1	0	0

Actually, if we look at the experimental results, we see that when the 1st and 10th experimental design points are replicated for 9 times, all replications are classified as Class I. In addition to that, the 11th experimental design point is classified as Class I, 8 times of the 9 replications. These best designs are among the tested ones, but we are seeking the best experimental design also among the ones which have not been tested before. For that purpose, we use optimization methods that are mentioned in Section 2.1.2.

Step 4: Find the optimal product/process design parameter settings for the desired mean and minimum variance

To obtain the optimal parameter settings for seeking Class I consistently, we first apply the ε-Constraint method, and then, the SNR method is used.

a) Obtain the optimal parameter settings by using ε-Constraint method

In order to develop the mathematical model required by this method, the expected value and the variance regression models are generated firstly by using Minitab 18 (2018). While the regression model is being performed, both main and interaction effects of the factors are considered. After an adequate starting point is obtained from stepwise regression, by adding the most significant factors and interactions or removing the least significant factors and interactions from the model we obtain more robust regression models. For the regression, factors A, B, C', D and E are set as continuous variables, since these are originally continuous. But, factor F (cleaning method) is categorical (nominal), since it has only three levels (None, CM_2, CM_3).

For all different Random Forests, i.e. weighted schemes, we obtain different expected value and variance models. The expected value and the variance models created for these Random Forests are given in Figure 3.9-3.12.

EV = -3.046+1.268A+1.146B-0.896C'+1.5877E+1.740F_2+1.050F_3
+0.4468C'*C'-0.1366A*B-0.3697C'*E+0.1479C'*F_2-0.2466C'*F_3
-1.3942E*F_2-0.4579E*F_3

R^2=99.92% R^2(adj)= 99.64% R^2(pred)= 98.57%

Var = -6.470+2.3549A+4.9774B+0.4653C'+3.5696D-4.0713E+0.0680F_2
+0.333F_3-1.3034B*B-0.4256D*D+0.8063E*E-0.9471A*D+0.1329B*D
-0.0390A*F_2+0.1150A*F_3+0.1509C'*F_2-0.2862C'*F_3

R^2=100.00% R^2(adj)= 99.93% R^2(pred)= 96.69%

Figure 3.9. Expected Value and Variance Regression Equations of RF_E for the Surface Defects Case

EV = 1.8872+0.0103A+0.0424B-1.3901C'-0.2119D-0.7302E+1.5424F_2
+0.6230F_3+0.45258C'*C'+0.39541A*B-0.29950A*C'+0.31552A*E
+0.28371D*E-0.2614D*F_2-0.1427D*F_3-0.48180E*F_2+0.1568E*F_3

R^2=100.00% R^2(adj)= 99.99% R^2(pred)= 88.91%

Var = 1.7842+0.8987A+2.6162B-1.4683C'-1.0229D-1.7512E+0.8124F_2
+0.1353F_3-0.57703B*B+0.38719E*E-0.61511A*B+0.19487A*C'
+0.38476B*E+0.63268C'*D-0.18262D*E-0.74259E*F_2-0.56201E*F_3

R^2=100.00% R^2(adj)= 99.99% R^2(pred)= 97.54%

Figure 3.10. Expected Value and Variance Regression Equations of RF_L for the Surface Defects Case

EV = -0.83+2.895A-2.666B-1.725C'+2.021E-0.701A*A+0.787B*B +0.725C'*C' +0.294A*B-0.681C'*E

R^2=95.42% R^2(adj)= 90.27% R^2(pred)= 77.75%

Var=0.271+0.317A-0.411E+0.529E*E-0.231A*B+0.319A*C'-0.546A*E -0.319C'*E

R^2=86.19% R^2(adj)= 76.52% R^2(pred)= 52.07%

Figure 3.11. Expected Value and Variance Regression Equations of RF_P for the Surface Defects Case

EV = -4.2211+1.76487A+0.84209B-0.3269C'+0.08647D+1.4759E-0.3944F_2 +0.6844F_3+0.55043C'*C'-0.72497E*E-1.02498A*C'+0.45911A*E +0.04529C'*E +0.28407D*E+0.32112B*F_2-0.12922B*F_3

R^2=100.00% R^2(adj)= 100.00% R^2(pred)= 99.99%

Var = -0.0775+2.0094A+0.11892B-0.1223C'-0.4385D-1.2727E-0.2976F_2 -0.3674F_3-0.45314A*A+0.04190D*D+0.46174E*E-0.11205A*D +0.24619C'*D-0.21497C'*E+0.15667B*F_2+0.1602B*F_3

R^2=100.00% R^2(adj)= 99.96% R^2(pred)= 99.69%

Figure 3.12. Expected Value and Variance Regression Equations of RF_X for the Surface Defects Case

To determine the best levels of parameters for the desired expected value and minimum variance, regression models for the expected value and the variance are used to formulate the RPD problem as a Multi Objective Optimization problem as given in Eqution 3.5. By using the ε-Constraint Method, minimization of variance is set as the objective function, and the expected value model is transformed into a constraint by selecting an upper bound such that the difference from the target value are less than or equal to ε. Also, the difference between the expected value and target value has added to the objective function with a very small coefficient.

For each Random Forest model, we construct a non-linear mathematical model that is based on their own expected value and variance models. Respectively, these models are as follows:

1. **For RF_E:**

$$Min\ \ (\overset{\wedge}{\text{Var}}) + (|\overset{\wedge}{\text{EV}}\text{-}1| \times 10^{-4})$$

s.t.

$$|\overset{\wedge}{\text{EV}}\text{-}1| \leq \varepsilon$$

$$\overset{\wedge}{\text{EV}} \geq 0$$

$$\overset{\wedge}{\text{Var}} \geq 0$$

$\overset{\wedge}{\text{EV}} = -3.046 + 1.268\times A + 1.146\times B - 0.896\times C' + 1.5877\times E + 0.0\times F_1 + 1.740\times F_2 + 1.05\times F_3 + 0.4468\times C'^2 - 0.1366\times A\times B - 0.3697\times C'\times E + 0.0\times C'\times F_1 + 0.1479\times C'\times F_2 - 0.2466\times C'\times F_3 + 0.0\times E\times F_1 - 1.3942\times E\times F_2 - 0.4579\times E\times F_3$

$\overset{\wedge}{\text{Var}} = -6.47 + 2.3549\times A + 4.9774\times B + 0.4653\times C' + 3.5696\times D - 4.0713\times E + 0.0\times F_1 + 0.068\times F_2 + 0.333\times F_3 - 1.3034\times B^2 - 0.4256\times D^2 + 0.8063\times E^2 - 0.9471\times A\times D + 0.1329\times B\times D + 0.0\times A\times F_1 - 0.039\times A\times F_2 + 0.115\times A\times F_3 + 0.0\times C'\times F_1 + 0.1509\times C'\times F_2 - 0.2862\times C'\times F_3$

$F_1 + F_2 + F_3 = 1$

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

$F_1, F_2, F_3 \in \{0, 1\}$

2. **For RF_L:**

$$Min \ (\hat{\text{Var}}) + (|\hat{\text{EV}}\text{-}1| \times 10^{-4})$$

$s.t.$

$$|\hat{\text{EV}}\text{-}1| \leq \varepsilon$$

$$\hat{\text{EV}} \geq 0$$

$$\hat{\text{Var}} \geq 0$$

$\hat{\text{EV}} = 1.8872 + 0.0103\times A + 0.0424\times B - 1.3901\times C' - 0.2119\times D - 0.7302\times E + 0.0\times F_1 + 1.5424\times F_2 + 0.6230\times F_3 + 0.45258\times C'^2 + 0.39541\times A\times B - 0.2995\times A\times C' + 0.31552\times A\times E + 0.28371\times D\times E + 0.0\times D\times F_1 - 0.2614\times D\times F_2 - 0.1427\times D\times F_3 + 0.0\times E\times F_1 - 0.4818\times E\times F_2 + 0.1568\times E\times F_3$

$\hat{\text{Var}} = 1.7842 + 0.8987\times A + 2.6162\times B - 1.4683\times C' - 1.0229\times D - 1.7512\times E + 0.0\times F_1 + 0.8124\times F_2 + 0.1353\times F_3 - 0.57703\times B^2 + 0.38719\times E^2 -0.61511\times A\times B + 0.19487\times A\times C' + 0.38476\times B\times E +0.63268\times C'\times D - 0.18262\times D\times E + 0.0\times E\times F_1 - 0.74259\times E\times F_2 - 0.56201\times E\times F_3$

$F_1 + F_2 + F_3 = 1$

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

$F_1, F_2, F_3 \in \{0, 1\}$

3. For RF_P:

$$Min\ (\hat{\mathrm{Var}}) + (|\hat{\mathrm{EV}}\text{-}1| \times 10^{-4})$$

s.t.

$$|\hat{\mathrm{EV}}\text{-}1| \le \varepsilon$$

$$\hat{\mathrm{EV}} \ge 0$$

$$\hat{\mathrm{Var}} \ge 0$$

$\hat{\mathrm{EV}}$ = -0.83 + 2.895×A - 2.666×B - 1.725×C' + 2.021×E - 0.701×A^2 + 0.787×B^2 + 0.725×C'^2 + 0.294×A×B - 0.681×C'×E

$\hat{\mathrm{Var}}$ = 0.271 + 0.317×A - 0.411×E + 0.529×E^2 - 0.231×A×B + 0.319×A×C'^2 - 0.546×A×E - 0.319×C'^2×E

A, B, C', E ≥ 0

4. **For RF_X:**

$$Min \ (\hat{\text{Var}}) + (|\hat{\text{EV}}\text{-}1| \times 10^{-4})$$

s.t.

$$|\hat{\text{EV}}\text{-}1| \leq \varepsilon$$

$$\hat{\text{EV}} \geq 0$$

$$\hat{\text{Var}} \geq 0$$

$\hat{\text{EV}}$ = -4.2211 + 1.76487×A + 0.84209×B - 0.3269×C' + 0.08647×D + 1.4759×E + 0.0×F_1 - 0.3944×F_2 + 0.6844×F_3 + 0.55043×C'^2 - 0.72497×E^2 - 1.02498×A×C' + 0.45911×A×E + 0.04529×C'×E + 0.28407×D×E + 0.0×B×F_1 + 0.32112×B×F_2 - 0.12922×B×F_3

$\hat{\text{Var}}$ = -0.0775 + 2.0094×A + 0.11892×B - 0.1223×C' - 0.4385×D - 1.2727×E + 0.0×F_1 - 0.2976×F_2 - 0.3674×F_3 - 0.45314×A^2 + 0.0419×D^2 + 0.46174×E^2 - 0.11205×A×D + 0.24619×C'×D - 0.21497×C'×E + 0.0×B×F_1 + 0.15667×B×F_2 + 0.1602×B×F_3

$F_1 + F_2 + F_3 = 1$

A, B, C', D, E ≥ 0

$F_1, F_2, F_3 \in \{0, 1\}$

We have used MATLAB/BARON (2013) software for solving these problems. According to different ε values, we have obtained different optimal solutions based on the expected value and the variance. These solutions are in Table 3.16.

Table 3.16. Solutions of ε-Constraint Method for the Surface Defects Cases

Solutions		A	B	C'	D	E	F	ε	$\hat{EV}$	$\hat{Var}$
1	RF_E	1.2440	1.0270	1.5140	3	1.7490	1	0	1	0.000002
2		1	1	2.9690	1	1	1	0.000003	0.999997	0.005000
3		1	1.9830	2.9960	1	2.9890	2	0.000004	1.000004	0.000020
4		1.2540	1.0330	1.4690	2.7300	1.7250	1	0.000031	1.000031	-0.000100
5	RF_L	1.0310	1.0110	1.9110	2.9820	1.9800	1	0	1	0.000020
6		1.0040	1.0240	2.9990	1.0060	1.0490	1	0.000006	1.000006	0.000110
7		1	1.9880	2.9970	1	3	2	0.00005	1.000050	0.000010
8		1.0400	1.007	1.9180	2.9860	1.9920	1	0.009970	1.009970	-0.000030
9	RF_P	1.0402	1.7764	3	-	2.8965	-	0	1	0.000050
10		1.0041	1.9136	2.9978	-	2.9004	-	0.000010	1.000010	-0.000040
11		1.1630	1	1.8245	-	2.1608	-	0.000010	0.999990	0.000020
12		1.1857	1.0111	2.7710	-	1.0053	-	0.000070	1.000070	0.000010
13	RF_X	1	1	2.0017	3	2.0155	1	0	1	0.0000001
14		1.0245	1.0705	1.9597	2.9428	1.9493	1	0.000010	0.999990	-0.000020
15		1.0326	1.0483	2.9569	1.0445	1.0605	1	0.000050	1.000050	-0.000020
16		1	2	3	1	3	2	0.00060	1.000600	-0.001800

As seen in Table 3.16, in some solutions, variance values that are calculated based on their regression models are estimated to be less than 0, but they are very small and very close to 0, so we can assume them to be equal to 0. And also for some solutions, some factors are not at their fixed levels, since we set these values as continuous. To find the original values of these factors, we can use a back transformation from these coded levels.

We have a chance to select the best parameter setting according to our purposes. So, if our main aim is to have the expected value very close to the target value, we can choose the 1st, 5th, 9th or 13th solutions since the expected values, which are calculated by using their expected value regression model, under these solutions are at the target value. Moreover, the expected values that are calculated by using Equation (3.2) after predicting probabilities from related Random Forests are 1, the target value, for these solutions. On the other hand, if our main purpose is to have the minimum variance, we can select most of the solutions.

As seen in Table 3.16, there are no significant differences between the results of the solutions for both the expected value and the variance. All RFs give the solution whose estimated expected value is equal to the target value, 1. Moreover, the estimated variance values of all RFs are very close or equal to 0. However, we can state that the estimated variance values of RF_X are slightly better than the others. So, since there is no significant superiority of a single method over another among all methods, at this stage, we can choose the Random Forest that has the best performance by considering the classification performance of the Random Forests. For that, we can take advantage of the performance measures mentioned in Step 2. For our class of interest, 1, RF_E and RF_X are the models which have satisfactory performance based on the class based comparison. For that reason, since optimization performances of all RF models are similar, and estimated variance values and the classification performance of RF_X are better than those of others, we can choose the exponentially-weighted Random Forest, RF_X, as the method. In addition to that, since the OOB error estimate is high in RF_X, RF_E can be chosen as an alternative model.

According to these ε-Constraint method solutions, D and F do not have a significant effect on the mean and variance for RF_P. So, we set their levels to any appropriate value, or to the most economical values.

b) Obtain the optimal parameter settings by using SNR

During regression modeling to find the best parameter settings, SNR is used in the second optimization method. For that purpose, ordinary least squares and response optimization (modified generalized reduced gradient) methods adopted by Minitab 18 (2018) are used to fit the model. SNR models are developed by stepwise regression and further iterations based on p-value. Also, by setting different starting points for the response optimizer, we try to achieve the best parameter settings.

Obtained optimal solutions based on SNR values for all Random Forests, weighted schemes, are given in Table 3.17. As it is seen from the table, since models have very high R-Sq, R-Sq(adj) and R-Sq(pred) values, we can say that these four models, i.e. weighted Random Forests, are highly adequate.

Table 3.17. Solution of the SNR model for the Surface Defects Case

Random Forests	**Regression Results**			**Optimizer Results**						$\hat{\text{SNR}}$
	R-Sq	R-Sq (adj)	R-Sq (pred)	A	B	C'	D	E	F	
RF_E	99.98%	99.87%	98.79%	1	1	2	3	1.990	1	0
RF_L	99.99%	99.95%	98.96%	1	1.720	3	1	3	1	0
RF_P	99.95%	99.70%	94.93%	1	1	2.010	3	2.003	1	-0.005
RF_X	99.97%	99.83%	98.66%	1	1	2	3	2.250	-	0

The results show that the estimated values of SNR are 0 (0 is the optimum level for this smaller the better case) for Random Forests RF_E, RF_L and RF_X. Although

R-Sq, R-Sq(adj) and R-Sq(pred) values of all models are also high, we can say that for RF_L, these values are slightly better than the others. Therefore, it can be logical to choose the linearly-weighted Random Forest, RF_L, since weights of other classes are not very low, either. In this way, the prediction power of other classes would also be high. Looking at both classification and optimization performances of the models, we see that RF_P has both the best OOB error estimate of the Random Forest and the OOB error rate of Class I, while the SNR value of it is slightly worse than the other models. On the other hand, SNR based optimization performances of the RF_L and RF_X are satisfactory, even though the classification performance of these are slightly worse compared with RF_P. If we look at it from another point of view, classification and SNR based optimization performances of the RF_E are also satisfactory, and it can also be considered of as an adequate model.

Solutions of RF_X has the insignificant factor, F. For that model, we can set any value to this insignificant factor, but, generally, the most economic values are preferred.

We try to find the best parameter settings to have Class I. For that we use two different optimization methods, ε-Constraint and SNR. In ε-Constraint method, the expected value and variance are taken into consideration separately. On the other hand, in the maximization of SNR method, the expected value and variance are combined into a single objective function, SNR. One can choose any of these methods.

Step 5: Confirmation of results and revisiting the problem

When we examine results of both methods, it can be stated that while some of their results are different from the tested ones, some of the others are very similar to the tested ones if we consider that the levels of the parameters are not very sensitive. Since we define some appropriate parameters as continuous during setting the models, and Phadke (1989) sets these as categorical, we have achieved similar results to his results with little differences.

Since we borrow the data from Phadke's study (1989), we do not have the chance to replicate the optimal solutions. For solutions that are different from the tested ones, different ways mentioned in Section 3.1 (Step 5) can be performed for confirmation. On the other hand, for the optimal solutions that are very similar to the tested ones, Phadke (1989) has run the parameter design for 9 times, and he has obtained 9 times Class I for two of them, and for the other one he has got 8 times Class I and 1 time Class II. We have found some optimal solutions with the estimated probabilities of the classes are 1,0,0,0 and 0, respectively. When we want to confirm our solution with Phadke's replications, we could not use Chi-square since some of the probabilities of the classes are equal to 0. For such cases, the literature suggests the Exact Multinomial Test. We have tested the null hypothesis that we established by using the R-studio (R Core Team, 2017), R-Studio can use Monte Carlo and Chi-Square methods to measure the distance between observed and expected frequencies. Since our case is not suitable to use Chi-square, we use Monte Carlo approach. However, the test is not working properly when 0 is entered in the related cells. We have entered values of these cells very close to 0 instead of 0. In this case, we have found p-value as 1. This means that fail to reject the null hypothesis, that is there is no difference between our result and Phadke's replications. Moreover, we have observed that the reliability of the test increases when the number of experiments is increased.

Our problem is a Smaller-the-Better type, so we are seeking the parameter designs such that the wafer has the minimum number of surface defects. And it means we are looking for the Class I with the highest probability and the expected value that is very close to 1 with minimum variance and maximum SNR values. As we expect, Random Forests whose weights of Class I are higher find the solution with higher SNR values. We can conclude that, giving weight to the classes -especially giving more weight to our class of interest- increases the prediction power of them, since their SNR values are higher than the unweighted Random Forest.

This example data set was also analyzed before by Karabulut (2013), and if we want to compare our results with her results, an exact comparison will not be possible, because she assumes that all factors are categorical. She gets as optimal solution $A_1B_1(C_1)C'_3E_1$ by using Logistic Regression model. She has also found that the estimated probability of Class I for that solution is 0.9498, the expected value is 1.0718, the variance is 0.1248 and the SNR value is -1.053. In our solutions, we obtain the parameter designs, whose expected values are 1, and also the estimated probability of Class I is 1 for solutions numbers 1, 5, 9, and 13 in Table 3.16 ($A_{1.244}B_{1.027}C'_{1.514}D_3E_{1.749}F_1$, $A_{1.031}B_{1.011}C'_{1.911}D_{2.982}E_{1.98}F_1$, $A_{1.0402}B_{1.7764}C'_2E_{2.9865}$, and $A_1B_1C'_{2.0017}D_3E_{2.0155}F_1$, respectively) which is solutions of ε-Constraint method. In addition to that, we obtain the variance values are lower than Karabulut's results for all given optimal solutions and SNR value as 0 for solutions $A_1B_1C'_2D_3E_{1.99}F_1$, $A_1B_{1.72}C'_3D_1E_3F_1$, and $A_1B_1C'_2D_3E_{2.25}$. A proper way of comparing the performances of the two approaches would be collecting data at the optimal solutions of these two approaches separately and then analyzing which of the two yields closer results to the observed mean and variance of the classes.

In Karabulut's (2013) study, for this problem, the best methods are Logistic Regression Model Optimization (LRMO), Accumulation Analysis Method (AA) and Weighted Signal to Noise Ratio Method (WSNR). All three methods give the same solution and the same SNR value. Scoring Scheme (SS) and Weighted Probability Scoring Scheme Method (WPSS) yield worse solutions compared to the other methods. Although Karabulut's study assumes variables as categorical and our results are similar to their results, this may indicate that Random Forest provides meaningful solutions. Accordingly if LR was to use variables as continuous similar results could have been derived. Although the results are not comparable in the sense that LR solutions assume variables as categorical and Random Forest results are based on continuous variables, the similarity of outcomes could indicate Random Forest is a meaningful alternative.

3.3 Illustrative Case Study II: Inkjet Printer Case

In this section, another example data set is used to apply our proposed method. This data set has been used by Logothetis (1992) and Karabulut (2013) for the RPD of an inkjet printer machine by using Accumulating Analysis, and for comparing five different RPD methods, respectively.

Step 1: Collecting data and overcoming the problems caused by missing values

This data set has been collected by Logothetis (1992) to optimize the blending of the ink mixture to use an inkjet printer by using Accumulation Analysis. There is a total of six substances (factors) affecting this ink mix; these are MeOH, dye, carbiton, PM, resin and water. MeOH is the most commonly used ingredient among these substances for the ink mixture. For this reason, firstly, the amount of each substance, except MeOH, is decided as the ratio of the whole mixture, and then, MeOH is used for the remaining part of the mix to complete the ratio of the ink mixture to 100%. Here, adhesive ability of the ink mixture for each experimental trial is tested, and parameter design is done to reach a high level of adhesiveness. In the experiment phase, ink is mixed according to the determined factor levels, then, with this mixture, a code prints five times on plastic and five times on metal substrates (here, plastic and metal substrates are noise factors). These prints are kept under the same environmental conditions for one night. After one night, in the test phase, adhesive ability of the ink mixture is evaluated by counting the number of rubbing the print over a suitable material until the code becomes unreadable. To sum up, values of all factors and quality characteristic are defined by Logothetis (1992) as presented in Table 3.18:

Table 3.18. Controllable Factors and Their Levels for the Inkjet Printer Case

Factors	**Levels**	
	0	1
A. Dye	1%	3%
B. Carbiton	1.5%	2.5%
C. PM	6.5%	9.5%
D. Resin	8%	12%
E. Water	10%	20%

If the printed code does not become to unreadable at the end of many rubs, the ink mixture is considered to be of a high quality. According to the number of rubs, the results are classified into four different groups, and related values are shown in Table 3.19. Here, the greater the range of rubs, the better the quality.

Table 3.19. Ranges for Numbers of Rubs for Each Class for the Inkjet Printer Case

Classes	**Range**	
	Min	Max
I	1	10
II	11	18
III	19	25
IV	26	∞

We are looking for an ink mixture for which the code is not readable after at least 26 rubs, so it is a larger-the-better type problem. Responses of the data are created in a sequential order; Class I refers to the smallest number of rubs and Class IV refers to the highest number of rubs to keep the prints in a still readable format. Class IV is the preferred class. In other words, while class labels increase, quality increases.

Eight different experiments were carried out with factor levels depending on the L_8 orthogonal array, and each experiment runs for 10 times. Finally, responses of these 10 runs for each experiment are classified according to the ranges for number of rubs. The numbers of observations for each class for the eight experiments are tabulated in Table 3.20.

Table 3.20. Experimental Design and Data Collected for the Inkjet Printer Case

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	8	1	0	1
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

Because there is no missing value in the data, no manipulation has been made for that purpose.

Step 2: Growing random forest trees and handling imbalanced data

The data collection experiment is designed so as to run 10 times for 8 different factor level combinations so we have 80 observations in our data set. This data set is analyzed with Random Forest algorithm by using the R-studio software (R Core Team, 2017).

First of all, Random Forest creates bootstrap subsamples each with a size of 80, which is equal to the size of the original data set, with replacement. Secondly, the numbers of attributes to use at each split are determined. As mentioned before in Section 2.2.1, this number is calculated by using $\lfloor \sqrt{p} \rfloor$, where p is the number of attributes, and for this problem, this number is $\lfloor \sqrt{5} \rfloor = 2$. Also, in R, the best split value is determined according to OOB error estimate, and the result is again equal to 2 as is the case with the formula above. The best split value graph in R is shown in Figure 3.13.

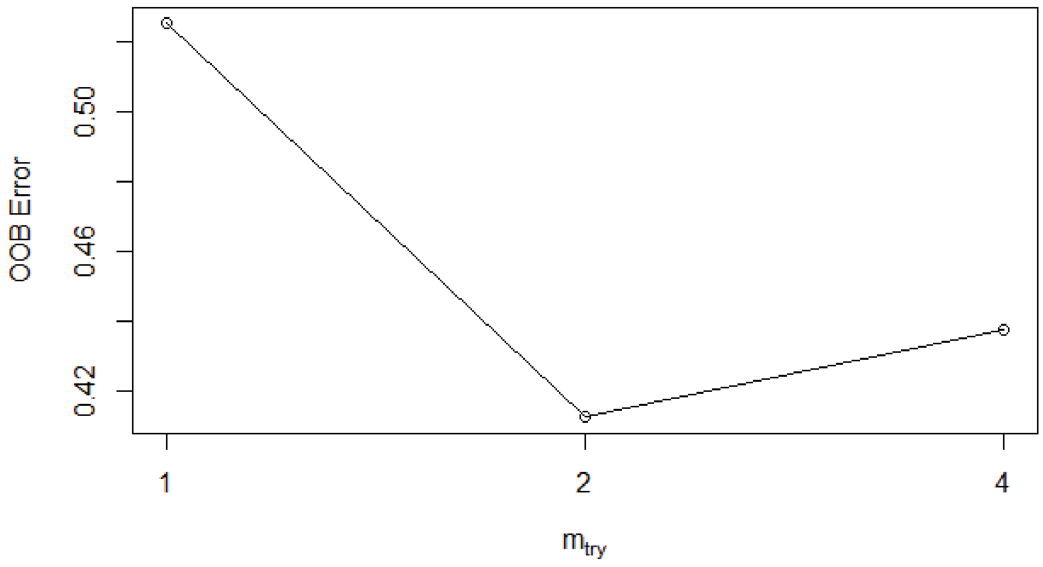


Figure 3.13. The Best Split Value versus OOB Error Rate for the Inkjet Printer Case

In addition, the best number of trees in the forest is tested with different values between 100-10000, and the best value is set as 200 according to both OOB error rate and consuming of time.

The data is not separated equally for all classes. Our class of interest, Class IV, has the second highest number of trials, and Class I, which is the least desired class, has the maximum number of trials. So, as we mentioned before, we are going to give weights to the classes according to four different strategies (Equally, Linearly, Piecewisely and

Exponentially distributed weights). Before giving the weights, firstly, we look at the distribution of trials for classes.

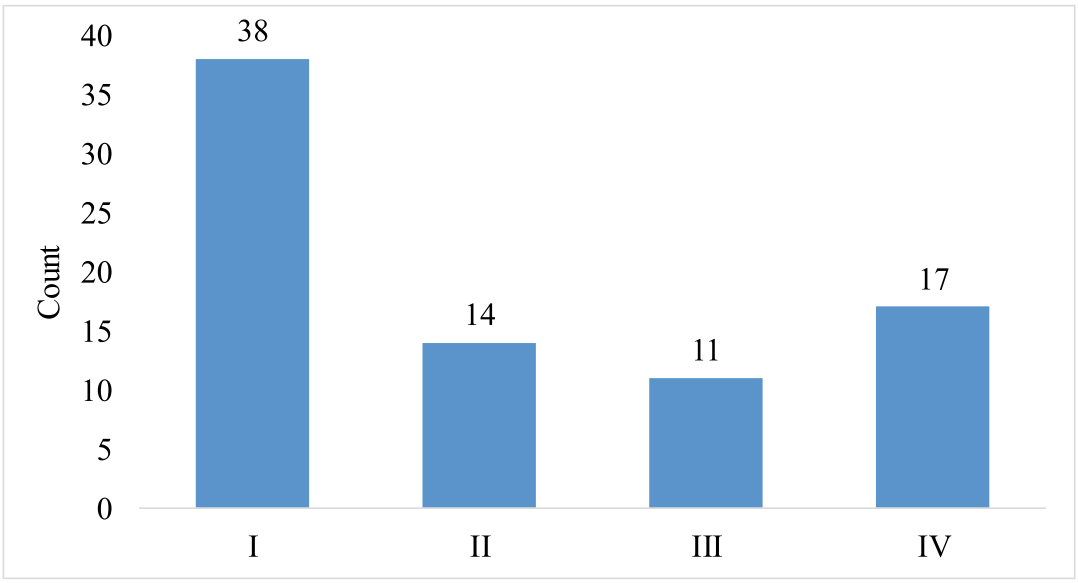


Figure 3.14. Distribution of the Data among Classes for the Inkjet Printer Case

When Figure 3.14 is observed, it is seen that there are 38 experiments for Class I, which is the least desired class, and this is more than twice the number for Class IV, which is our class of interest. The other classes (Class II and Class III) have fewer experiments than both Class I and Class IV. Briefly, these classes form 48%, 17%, 14% and 21% of the data set, respectively.

Weights based on specified strategies are given in Table 3.21 with their abbreviations. Here, our problem is a Larger-the-Better type problem, and our class of interest is Class IV. So, we give higher weights to Class IV.

Table 3.21. Class Weights of the Random Forests for the Inkjet Printer Case

Weighting Strategies	**RF Abbreviation**	**Weights of Classes**			
		I	II	III	IV
Equal Weights	RF_E	0.25	0.25	0.25	0.25
Linear Weights	RF_L	0.10	0.20	0.30	0.40
Piecewise Linear Weights	RF_P	0.166	0.167	0.167	0.500
Exponential Weights	RF_X	0.03	0.09	0.24	0.64

According to these weights, OOB error rate graphs of the Random Forests formed with 200 trees and 2 as the split value are as given in Figures 3.15-3.18.

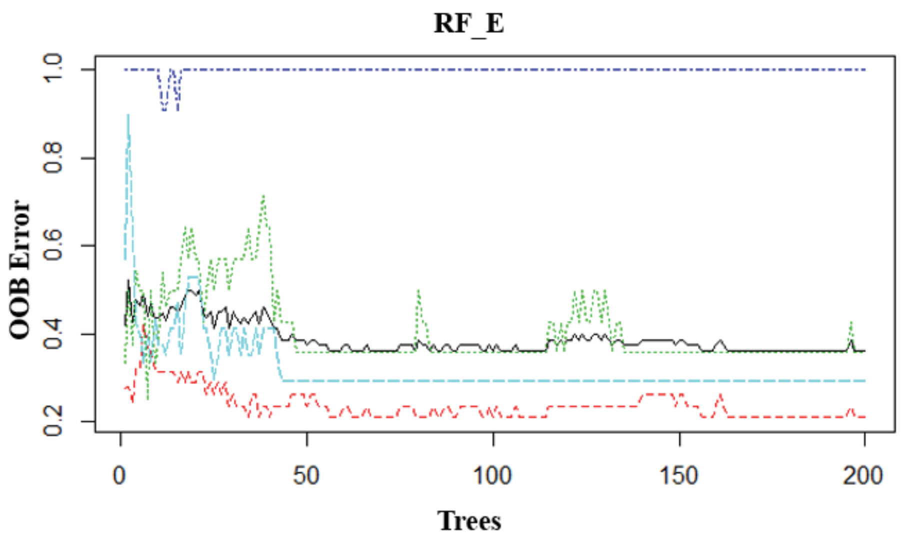


Figure 3.15. OOB Error Rate Graph of RF_E for the Inkjet Printer Case

According to the graph, OOB error estimate line for the data set is the black line, which is 36.25% on the average. Furthermore, the cyan line is for Class IV with an average of 29% OOB error rate, and the red line is for Class I with 21% OOB error rate on the average. Class I has the minimum OOB error rate. The green line signs Class II with an average of 35% OOB error rate. In addition, the class that has the maximum OOB error rate is Class III, and its error rate is 100% (blue line) on the average. As seen from the graph, this Random Forest can make satisfactory predictions for Class I, the least desired class, and Class IV, the class of interest. For Class II, prediction performance of the Random Forest might be sufficient, but for Class III, it is not the case. This Random Forest predicts all experiments for Class III incorrectly.

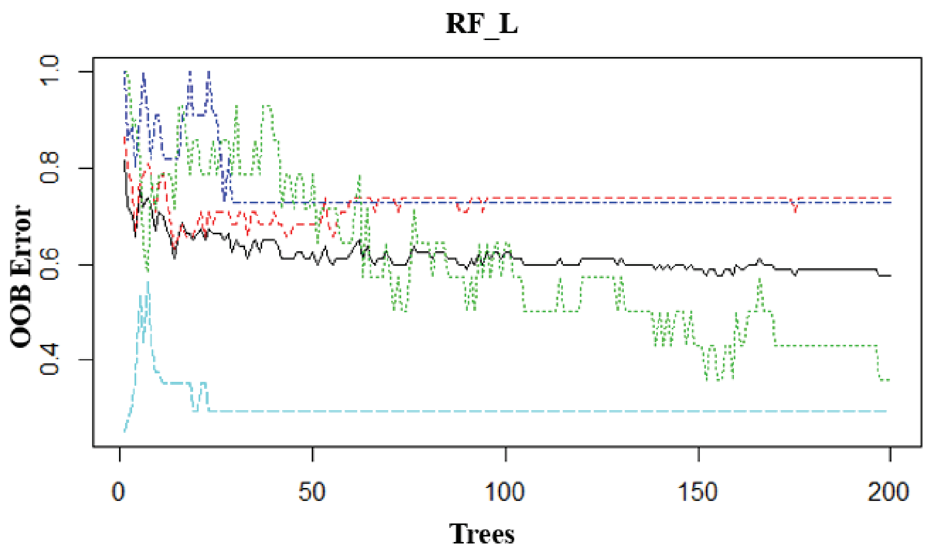


Figure 3.16. OOB Error Rate Graph of RF_L for the Inkjet Printer Case

We see that the OOB error estimate of the Random Forest increase (black line) from 36.25% to 57.5%. Also, average OOB error rates of Class I increase to 73% (red line). The OOB error rates of Class IV (cyan line), and Class II (green line) has not changed (29% and 35%, respectively). On the other hand, average OOB error rates of Class III decrease from 100% to 73% (blue line). So, we conclude that giving linearly increasing weights does not affect the prediction accuracy of Class IV and Class II, but affects those of the other classes. While the average OOB error rate of Class I increases, the average OOB error rate of Class III decreases. Our class of interest is

Class IV, and the class closest to our class of interest is Class III. Decreasing error rate of Class III can be evaluated as a better result.

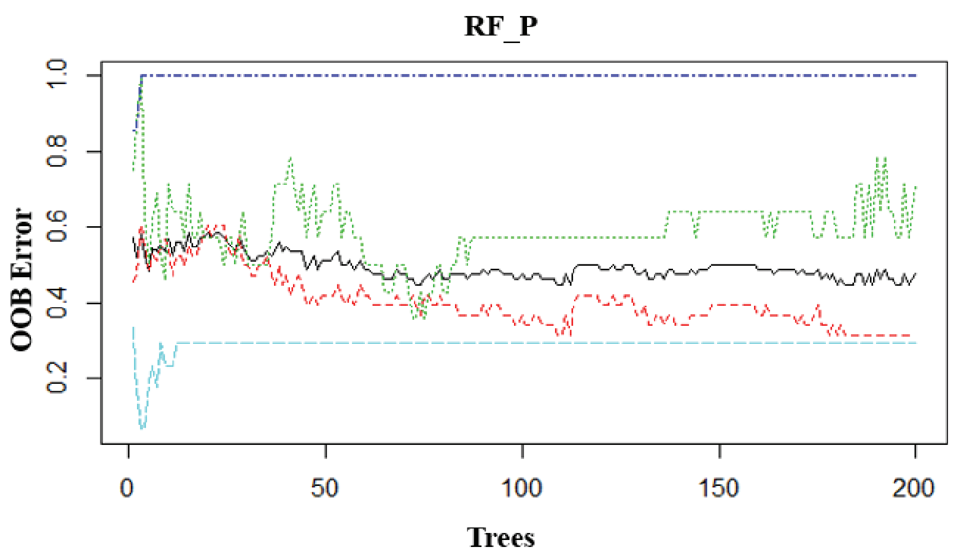


Figure 3.17. OOB Error Rate Graph of RF_P for the Inkjet Printer Case

When Figure 3.20. is analyzed, we understand that the average OOB error estimate of RF_P (black line- 50%) is less (better) than that of RF_L and higher (worse) than that of RF_E. The OOB error rate of Class IV still has not changed (cyan line - 29%). The OOB error rates of Class II and Class III have increased (green and blue line). The OOB error rate of Class II is 71% and that of Class III is 100%. Lastly, the red line represents Class I, and its OOB error rate has decreased to 31% on the average. So, in general, this Random Forest performs better than RF_L, but worse than RF_E. This Random Forest performs similar to RF_E and RF_L for our class of interest; that is, Class IV.

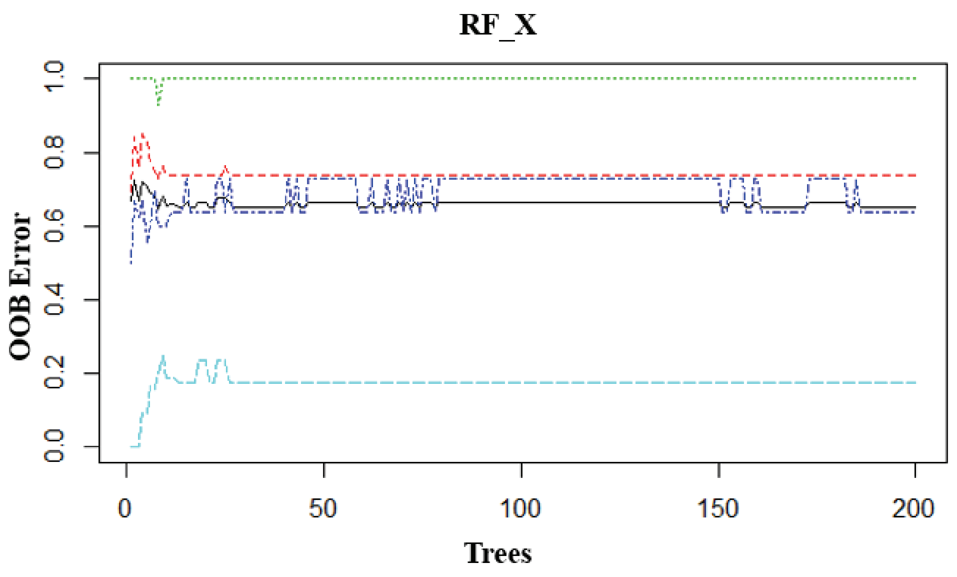


Figure 3.18. OOB Error Rate Graph of RF_X for the Inkjet Printer Case

RF_X is the Random Forest with the maximum OOB error estimate (65% - black line). As we expected, the average OOB error rate of Class IV in RF_X (cyan line – 17%) is the minimum among all constructed Random Forests; namely RF_E, EF_L, EF_P, and RF_X, because we have given 64% weight to Class I, and the remaining weights are distributed exponentially among the classes according to a increasing class order. The average OOB error rate of the other classes are very high and the rates are 74%, 100% and 63% for Class I, Class II and Class III, respectively.

In brief, when we evaluate OOB error rate graphs of these four different weighted Random Forests, including the unweighted Random Forest, RF_E has the minimum average OOB error estimate (36.25%). Average OOB error estimate of the other Random Forests are 57.5%, 47.5%, and 65% for RF_L, RF_P, and RF_X, respectively. RF_X is the Random Forest the OOB error rate of which for Class IV is the minimum (17%). Also, the average OOB error rate of Class III, which is the class closest to our class of interest, is minimum (63%) in RF_X. For other Random Forests, the average OOB error rate of Class IV is equal (29%) and higher than in RF_X. On the other hand, the Random Forest with the maximum average OOB error estimate is

RF_X. According to our purpose, we can choose any one of these Random Forests. We can select RF_X, if prediction performance of our class of interest (Class IV), and the closest class to Class IV is more important. However, if the general performance of the Random Forest is crucial to us, we can select RF_E. As an alternative, we can choose the appropriate weighted strategy after optimization.

Additionally, we can analyze these Random Forests as a classification method based on performance measures mentioned in Section 2.2.1.6. Furthermore, we can compare Random Forest and Logistic Regression by using these performance measures. Firstly, we examine the confusion matrices of the RFs and Logistic Regression in Tables 3.22-3.26.

Table 3.22. Confusion Matrix of RF_E for the Inkjet Printer Case

		Predicted				
	Classes	I	II	III		Sum
Actual	I	30	5	0	3	38
	II	4	9	0	1	14
	III	4	3	0	4	11
	IV	2	3	0	12	17
	Sum	40	20	0	20	80

Table 3.23. Confusion Matrix of RF_L for the Inkjet Printer Case

		Predicted				
	Classes	I	II	III	IV	Sum
Actual	I	10	5	12	11	38
	II	0	9	3	2	14
	III	0	3	4	4	11
	IV	0	3	1	13	17
	Sum	10	20	20	30	80

Table 3.24. Confusion Matrix of RF_P for the Inkjet Printer Case

		Predicted				
	Classes	I	II	III	IV	Sum
Actual	I	26	5	4	3	38
	II	1	9	3	1	14
	III	1	3	3	4	11
	IV	2	3	0	12	17
	Sum	30	20	10	20	80

Table 3.25. Confusion Matrix of RF_X for the Inkjet Printer Case

		Predicted				
	Classes	I	II	III	IV	Sum
Actual	I	10	0	7	21	38
	II	0	0	7	7	14
	III	0	0	5	6	11
	IV	0	0	1	16	17
	Sum	10	0	20	50	80

Table 3.26. Confusion Matrix of Logistic Regression for the Inkjet Printer Case

		Predicted				
	Classes	I	II	III	IV	Sum
Actual	I	33	2	0	3	38
	II	8	5	0	1	14
	III	6	1	0	4	11
	IV	3	2	0	12	17
	Sum	50	10	0	20	80

Confusion matrices are constructed based on testing all of the data set, meaning that there is no test set. Since we use all of the data for learning, do not split them into train and test sets.

Measures that are calculated based on these confusion matrices are tabulated in Table 3.27.

When we look at the table, we see that the accuracy of Logistic Regression is worse than that of RF_E, and equal to that of RF_P, but better than those of RF_L and RF_X. But, as we mentioned before, since our data is imbalanced, we should use balanced values. And all balanced accuracies of weighted Random Forests are better than those of the regular Random Forest. As we expected, RF_X, for which we give the highest weight to Class IV, has the highest balanced accuracy. Also, balanced accuracy based Kappa increases as the weight of Class IV increases. For F- Score and G-Mean, while some Random Forests show better performance than Logistic Regression, some of them show similar performance. AUC values of all Random Forests (except for RF_X) are also better than that of Logistic Regression.

In addition to these comparisons, class-based measures can be compared as well. The class-based comparison table is given in Table 3.28. Here, we see that, for our class of interest (Class IV), the method that shows the best performance according to Recall is RF_X, and the methods that show the best performance according to Precision and Specificity are RF_E, RF_P and LR. We understand that, by changing the weights, we can increase or decrease the performance of a Random Forest. When we evaluate all these performance measures, the best performing method varies according to each performance measure. However, all Random Forests generally perform better than Logistic Regression. It may be misleading to choose the best method in this stage. But, these performance measures can also be used as an indicator in the optimization step while deciding on the method to be selected.

.

Table 3.27. Comparison of Random Forests and Logistic Regression According to Performance Measures for the Inkjet Printer Case

Random Forests	**Acc**	**Balanced Acc**	**Kappa**	**Balanced Kappa**	**F-Score**	**G-Mean**	**AUC**
RF_E	64%	54%	46%	30%	60%	0%	73%
RF_L	45%	54%	28%	45%	44%	62%	68%
RF_P	63%	62%	47%	46%	63%	69%	75%
RF_X	39%	72%	21%	64%	34%	0%	61%
LR	63%	48%	40%	48%	57%	0%	70%

If we want to analyze the importance of attributes/factors, we see that the most important factors are A and B for both measures (Mean Decrease Accuracy and Mean Decrease Gini). On the other hand, the least important factor is generally D. There are some slight differences between variable importance orders of Random Forests, but generally, the most and the least important attributes are the same. All variable importance charts are given in Appendix D1. We can conclude that in the case where some factors have to be chosen for the trials, these important factors can be selected firstly.

Step 3: Estimating the probability of each class, and calculating expected value and variance for each experiment

The expected value, the variance and the SNR value (Formula 2.6. is used for this problem, since this is a larger-the-better problem) are calculated based on the estimated probabilities for each constructed Random Forest, and the results are given in Appendix D.2.

For all Random Forest trees, except for RF_E, the eighth experiment, $A_1B_1C_1D_0E_1$, is the best design among the tested ones, according to the probability of our class of interest, the expected value, the variance, and SNR. For RF_E, the seventh experimental design, $A_1B_1C_0D_1E_0$, is the best experiment among the tested ones. At the seventh and eighth parameter designs, the probability of Class IV is at its highest value, the expected value is very close to the desired value which is equal to 4.

And also for the eighth parameter design, the variance is minimum in RF_L, RF_P, and RF_X. But in RF_E, the best design based on the probability of our class of interest, the expected value and the SNR is not the best design in respect of the variance. Best parameter designs among the ones tested for all Random Forest applications, and their probabilities of classes, expected values, variances, and SNRs are given in Table 3.29.

Table 3.28. Comparison of Random Forest and Logistic Regression According to Class-Based Performance Measures

Classes	**Recall**					**Precision**					**Specificity**				
	RF_E	RF_L	RF_P	RF_X	LR	RF_E	RF_L	RF_P	RF_X	LR	RF_E	RF_L	RF_P	RF_X	LR
I	79%	26%	68%	26%	87%	75%	100%	87%	100%	66%	76%	100%	90%	100%	60%
II	64%	64%	64%	0%	36%	45%	45%	45%	0%	50%	83%	83%	83%	NA	92%
III	0%	36%	27%	45%	0%	0%	20%	30%	25%	0%	100%	77%	90%	78%	NA
IV	71%	76%	71%	94%	71%	60%	43%	60%	32%	60%	87%	73%	87%	46%	87%

Table 3.29. Best Parameter Design Obtained from Different Random Forests for the Inkjet Printer Case

Random Forests	**Factors**					**Estimated Probabilities from Random Forests**				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	A	B	C	D	E	$\hat{P}(Y = I)$	$\hat{P}(Y = II)$	$\hat{P}(Y = III)$	$\hat{P}(Y = IV)$			
RF_E	1	1	0	1	0	0.1	0	0.10	0.80	3.6000	0.8400	10.3544
RF_L	1	1	1	0	1	0	0	0.03	0.97	3.9700	0.0291	11.9518
RF_P	1	1	1	0	1	0	0	0.01	0.99	3.9900	0.0099	12.0114
RF_X	1	1	1	0	1	0	0	0.01	0.99	3.9900	0.0099	12.0114

Step 4: Find the optimal product/process design parameter settings for desired mean and minimum variance

In this problem, we want to achieve the maximum range for the number of rubs. In other words, we seek a parameter setting with the estimated probability of Class IV at its highest value, the expected value close or equal to 4 and minimum variance. To solve the problem, as mentioned before in Section 3.1, ε-Constraint and SNR methods are applied.

a) Obtain the optimal parameter settings by using ε-Constraint method

To obtain the maximum range for the rubs, firstly, the ε-Constraint method is used. To this end, the expected value and the variance regression models that are used for the mathematical model are obtained from Minitab 18 (2018). For regression models, firstly stepwise regression is performed and then by adding the most significant factors and interactions or removing the least significant factors and interactions from the model we obtain more robust regression models. While regression models are being generated, both main effects of factors and their interactions are taken into consideration. Moreover, all factors are set as continuous since these are originally continuous. Since we have four different Random Forests, we get four different expected value and variance models. These models are shown in Figure 3.19-3.22:

$$\textbf{EV} = 1.593-0.412A-0.183B+0.568C-0.192E+2.665A*B-0.670A*C$$

$$\mathbf{R^2=99.57\%} \quad \mathbf{R^2(adj)=97.01\%} \quad \mathbf{R^2(pred)=72.67\%}$$

$$\textbf{Var} = 0.5686+0.0133A+0.2400B+0.0600C+0.2507A*B+0.0331A*C+0.1282A*E$$

$$\mathbf{R^2=99.91\%} \quad \mathbf{R^2(adj)=99.37\%} \quad \mathbf{R^2(pred)=94.24\%}$$

Figure 3.19. Expected Value and Variance Regression Equations of RF_E for the Inkjet Printer Case

EV = 2.861-0.917A-0.180B-0.297C-0.055E+1.910A*B+0.775A*E

R^2=96.97% R^2(adj)=78.81% R^2(pred)=0.00%

Var = 0.1815+1.115A+0.5717B-0.2501C+0.9235E-1.675A*B-0.871A*E

R^2=99.66% R^2(adj)=97.61% R^2(pred)=78.11%

Figure 3.20. Expected Value and Variance Regression Equations of RF_L for the Inkjet Printer Case

EV = 2.7038+0.1963A+0.4888B-0.1237D+0.5913A*B-0.2188A*C

R^2=98.83% R^2(adj)=95.92% R^2(pred)=81.35%

Var = 0.9077-0.1528A-0.2039B-0.2117C-0.5315A*B-0.3474A*D

R^2=97.90% R^2(adj)=92.65% R^2(pred)=66.42%

Figure 3.21. Expected Value and Variance Regression Equations of RF_P for the Inkjet Printer Case

EV = 3.216+0.351B+0.294E+0.394A*B-0.313A*C

R^2=92.67% R^2(adj)=82.895% R^2(pred)=47.85%

Var = 0.6131-0.1187B+0.2603D-0.4386A*B+0.1428A*C

R^2=94.42% R^2(adj)=86.99% R^2(pred)=60.35%

Figure 3.22. Expected Value and Variance Regression Equations of RF_X for the Inkjet Printer Case

To find the best level of parameters, which provides an expected value equals to 4 and minimum variance, these expected value and variance regression models are used to formulate the RPD problem as a Multi Objective Optimization problem using the ad hoc ε-Constraint Method model for four Random Forests. For solving these non-linear models, MATLAB/BARON (2013) is used. Here, again, the variance is minimized, but differently from the problem in Case Study-I, the expected value is tried to be maximized (because of larger-the-better problem), and the aimed target value is 4. As a result, in our ε-Constraint method, the objective function is to minimize the variance (and also to get a more robust solution; the difference between the expected value and the target value is added to the objective function after being multiplied by a very small coefficient); and the constraint is the difference between the expected value and target value (4) being in the ε neighborhood. The non-linear mathematical models for the Inkjet Printer Case are as follows:

1. **For RF_E:**

$$
\begin{aligned}
&Min\ \ (\overset{\wedge}{\mathrm{Var}}) + (|\overset{\wedge}{\mathrm{EV}} - 4| \times 10^{-4}) \\
&s.t. \\
&|\overset{\wedge}{\mathrm{EV}} - 4| \le \varepsilon \\
&\overset{\wedge}{\mathrm{EV}} \ge 0 \\
&\overset{\wedge}{\mathrm{Var}} \ge 0
\end{aligned}
$$

$$\overset{\wedge}{\mathrm{EV}} = 1.593 - 0.412\times A - 0.183\times B + 0.568\times C - 0.192\times E + 2.665\times A\times B - 0.67\times A\times C$$

$$\overset{\wedge}{\mathrm{Var}} = 0.5686 + 0.0133\times A + 0.24\times B + 0.06\times C + 0.2507\times A\times B + 0.0331\times A\times C + 0.1282\times A\times E$$

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

2. **For RF_L:**

$$Min\ (\hat{\mathrm{Var}}) + (|\hat{\mathrm{EV}} - 4| \times 10^{-4})$$
s.t.
$$|\hat{\mathrm{EV}} - 4| \le \varepsilon$$
$$\hat{\mathrm{EV}} \ge 0$$
$$\hat{\mathrm{Var}} \ge 0$$

$$\hat{\mathrm{EV}} = 2.861 - 0.917\times A - 0.18\times B - 0.297\times C - 0.055\times E + 1.91\times A\times B + 0.775\times A\times E$$

$$\hat{\mathrm{Var}} = 0.1815 + 1.115\times A + 0.5717\times B - 0.2501\times C + 0.9235\times E - 1.675\times A\times B - 0.871\times A\times E$$

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

3. **For RF_P:**

$$Min\ (\hat{\mathrm{Var}}) + (|\hat{\mathrm{EV}} - 4| \times 10^{-4})$$
s.t.
$$|\hat{\mathrm{EV}} - 4| \le \varepsilon$$
$$\hat{\mathrm{EV}} \ge 0$$
$$\hat{\mathrm{Var}} \ge 0$$

$$\hat{\mathrm{EV}} = 2.7038 + 0.1963\times A + 0.4888\times B - 0.1237\times D + 0.5913\times A\times B - 0.2188\times A\times C$$

$$\hat{\mathrm{Var}} = 0.9077 - 0.1528\times A - 0.2039\times B - 0.2117\times C - 0.5315\times A\times B - 0.3474\times A\times D$$

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

4. **For RF_X:**

$$Min\ (\hat{\text{Var}}) + (|\hat{\text{EV}} - 4| \times 10^{-4})$$

s.t.

$$|\hat{\text{EV}} - 4| \leq \varepsilon$$

$$\hat{\text{EV}} \geq 0$$

$$\hat{\text{Var}} \geq 0$$

$$\hat{\text{EV}} = 3.216 + 0.351 \times B + 0.294 \times E + 0.394 \times A \times B - 0.313 \times A \times C$$

$$\hat{\text{Var}} = 0.6131 - 0.1187 \times B + 0.2603 \times D - 0.4386 \times A \times B + 0.1428 \times A \times C$$

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

We solve these four mathematical model by using MATLAB/BARON (2013). Each model gives a different solution according to different ε values, as we expected. We present some of the best solutions of the RFs according to expected value and variance in Table 3.30.

As seen in Table 3.30, for some solutions, some factors are not at their fixed levels, since we set all values as continuous. To find the original values of these factors, we can use a back transformation from these coded levels. Furthermore, variance values, which are calculated by using their variance regression models, of some solutions are estimated to be less than 0, but they are very small and very close to 0, so we can assume them to be equal to 0.

Table 3.30. Solutions of ε-Constraint Method for the Inkjet Printer Case

Solutions		A	B	C	D	E	$\boldsymbol{\varepsilon}$	$\hat{\text{EV}}$	$\hat{\text{Var}}$
1	RF_E	1	1	0	-	0	0.3370	3.6630	1.0726
2		1	0.9746	0	-	0	0.4000	3.600	1.0602
3		1	1	1	-	1	0.6310	3.3690	1.2939
4	RF_L	0.9975	0.9966	0.9839	-	0.8726	0	4	5.4×10^{-8}
5		0.9977	0.9932	0.9843	-	0.8109	0.0500	3.9500	-6.1×10^{-7}
6	RF_P	1	1	0	0	-	0.0198	3.9802	0.0195
7		1	1	0	0.0561	-	0.0267	3.9733	-1.4×10^{-8}
8		0.9050	0.9842	0.4504	0	-	0.2000	3.8000	-5×10^{-7}
9	RF_X	1	1	0	0	0.1327	0	4	0.0558
10		1	1	0	1	0	0.0390	3.9610	0.3161

We have been able to achieve better results by using the weighting property provided by the Random Forest method. We have obtained highly adequate results regarding both the expected value and the variance in the solutions of all weighted Random Forests. We obtain the solutions the closest to the target value from RF_L and RF_X, but obtained those the furthest to the target value from RF_E. On the other hand, while the solutions with minimum variance are obtained from RF_L, RF_P, and RF_X, the solutions with maximum variance are yielded by RF_E. It is clear that giving more weights to our class of interest allows obtaining better results compared to unweighted Random Forest.

For both the expected value and the variance, the worst results are obtained from the equal-weighted Random Forest, RF_E. As seen in Table 3.30, 1st, 2nd, and 3th solutions are the worst solutions for the variance. For the expected value, results of weighted Random Forests are generally better than those of the unweighted Random Forest. RF_L and RF_X models obtain epsilon as 0. But, for RF_E, the best value of epsilon is 0.337. Therefore, we can suggest choosing one of the weighted Random Forests to obtain satisfactory results regarding both expected value and variance. For the expected value, there are no significant differences between the results of RF_L and RF_X and also RF_P. On the other hand, it is seen that the variance values of RF_L and RF_P solutions are minimum. To decide on selecting the suitable model, we can also consider the classification performance of the Random Forest mentioned in Step 2. According to these performance measures, general and class based performance of RF_P is slightly better than those of RF_L and RF_X. For that reason, since there are no significant differences between optimization performances of the models, and since RF_P is superior to the others in the respect of variance we can choose RF_P. On the other hand, since both optimization and classification of RF_L and RF_X are satisfactory, RF_L and RF_X can also be chosen.

As in the Case Study-1, there are insignificant factors, notated by D for RF_E and RF_L while E for RF_P. These factor does not have a significant effect on the mean and variance. Therefore, we can set their levels to any appropriate value, or to the most economical value.

b) Obtain the optimal parameter settings by using SNR

To find the best parameter design, we use SNR method as the second way. For this purpose, we use ordinary least squares and response optimization (modified generalized reduced gradient) methods via Minitab 18 (2018) by considering both p-value and stepwise regression. Different starting points for response optimizer are also tried to achieve the best parameter settings. Since we have four different methods, there are four corresponding different SNR solutions. Solutions obtained from these different Random Forest models are given in Table 3.31.

As it is seen from Table 3.31, each Random Forest gives a different solution. The solutions of RF_L and RF_X are similar to each other. Since these models have very high R-Sq, R-Sq(adj) and R-Sq(pred) values (R-Sq(pred) value of RF_X is not high enough), we can say that they exhibit quite sufficient performance to fit the regression model.

Table 3.31. Solution of the SNR model for the Inkjet Printer Case

Random Forests	**Regression Results**			**Factors**					$\hat{SNR}$
	R-Sq	R-Sq (adj)	R-Sq (pred)	A	B	C	D	E	
RF_E	99.51%	96.56%	68.53%	1	1	0	-	0	10.6544
RF_L	99.27%	94.87%	53.08%	1	1	0.889	-	1	12.0273
RF_P	99.85%	99.48%	97.64%	1	1	0.889	0	-	12.0311
RF_X	81.53%	56.91%	0.00%	0.990	0.996	0.999	-	0.986	12.0410

The SNR values of the weighted Random Forests are better than that of the regular Random Forest, as we expected. As we can guess, the Random Forests for which we gave more weight to Class IV estimate this class better compared to the unweighted Random Forest. As can be seen in the table, as the weight of Class IV increases, SNR value increases as well. Looking at the classification performance and OOB error graphs, RF_P and RF_X seem to be preferable alternatives to solve this problem with SNR method. This example data set also verifies us and the other researchers who argue that giving weight to a class in Random Forest is a very useful and important property, and that this is very easy to use. In addition, this property is unique for Random Forest.
D is the insignificant factor for RF_E, RF_L, and RF_X while E is insignificant factor for RF_P. We can assign any value to these factors, but generally, the most economic levels are preferred.

In our study, two different optimization methods, ε-Constraint and SNR methods are used to find the best parameter settings to have Class IV for the Inkjet Printer Case. As we mentioned before, while in the SNR method, the expected value and variance are combined into a single objective function, SNR, in ε-Constraint method, the expected value and variance are taken into consideration separately. One can choose any of these methods.

Step 5: Confirmation of results and revisiting the problem

When we investigate results of both methods, it can be indicted that while some of their results are different from the tested ones, some of the others are very similar to the tested ones if we consider that the levels of the parameters are not very sensitive. Since we define some suitable parameters as continuous during setting the models, but Logothetis (1992) sets these as categorical, we have achieved similar results to his results with little differences.

Since we quote the data from Logothetis's study (1992), we do not have the chance to replicate the optimal solutions. For solutions that are different from the tested ones, different ways mentioned in Section 3.1 (Step 5) can be performed for confirmation. On the other hand, for the optimal solutions that are very similar to the tested ones, Logothetis (1992) has run the parameter design for 10 times, and he has obtained 6 times Class IV, 3 times Class III, and 1 time Class I for one of them, and for the other one he has got 6 times Class IV, 2 times Class I and 1 time Class II and III. Like Case Stuy-I we apply Exact Multinomial Test to confirm our results with Logothetis's replications. During the test R-studio (R Core Team, 2017), which use Monte Carlo approach, is used. For aforementioned experimental designs, we have found p-value as 0.67 and 0.65, respectively. This means that there are differences between our results and Logothetis's replications but we can accept these differences because as we mention in Section 3.1, we expect that probabilities could not to be exactly equal to each other. Moreover, p-value we obtain is at the acceptable level.

Our problem is a Larger-the-Better type, so we are looking for an ink mixture such that following the printing process, the print is still readable after being rubbed for more than 26 times. Therefore, our interest class is Class IV. So, we are seeking the parameter design that gives the highest probability of Class IV. And also, in parallel with this, the design yields an expected value equal or close to 4, while the variance is minimum and the SNR value is maximum.

Karabulut (2013) also tried to find the best parameter settings for this problem by using Logistic Regression method to determine the relationship between factors and outputs, and set the ANOVA model for the best parameter settings by assuming that all factors are categorical, i.e. she set all factors to their fixed levels, which are 0 and 1. At the end of her study, she finds that the optimal solution is as $A_1B_1C_0$ and the estimated probability for Class IV 0.6836 while the expected value, variance and SNR value are 3.4915, 0.7359 and 10.1379, respectively. It is expected that our solutions would differ from Karabulut's solution, because while she set all factors as categorical, we set them as continuous, since they are originally continuous. Generally, in our solution, especially in weighted Random Forests, estimated probability of Class IV, the expected value, the variance, and SNR values are all better than those included in Karabulut's solution. In our ε-Constraint solutions, which is given in Table 3.30, we find out the estimated probability of Class IV to be 1, the expected value to be 4 and the variance to be 0 in solution numbers 4 and 9. These solutions are $A_{0.9975}B_{0.9966}C_{0.9839}E_{0.8726}$ and $A_1B_1C_0D_0E_{0.1327}$. Furthermore, all SNR values we obtain are better (10.6544, 12.0273, 12.0311, and 12.0410 for RF_E, RF_L, RF_P, and RF_X, respectively) than Karabulut's results. These solutions are $A_1B_1C_0E_0$, $A_1B_1C_{0.889}E_1$, $A_1B_1C_{0.889}D_0$, and $A_{0.99}B_{0.996}C_{0.999}E_{0.986}$. A convenient way to compare the performance of the two approaches would to collect data at the optimal solutions of these two approaches and then analyzing which of the two achieve closer results to the observed mean and variance.

In addition, Karabulut (2013) found for this problem that all methods she used (LRMO, AA, WSNR, SS, and WPSS) give the same solution as the Logistic Regression as the optimal one, and she concluded that all these methods show the same performance. Although Karabulut's study assumes variables as categorical and our results are similar to their results, this may indicate that Random Forest provides meaningful solutions. Accordingly if LR was to use variables as continuous similar results could have been derived. Although the results are not comparable in the sense that LR solutions assume variables as categorical and Random Forest results are based on continuous variables, the similarity of outcomes could indicate Random Forest is a meaningful alternative.

3.4 Illustrative Case Study III: Duplicator Case

For Case Study III, our proposed method is applied to a data set that has been used by Logothetis and Wynn (1994) to determine the best condition for the paper feeding phase of a duplicator machine at a high speed by using Accumulating Analysis. The data set is also used by Karabulut (2013) to compare different RPD methods.

Step 1: Collecting data and overcoming the problems caused by missing values

In this case, there is a duplicator, and paper sheets feed it at a high speed, and it is intended that the best operating conditions to provide the accomplished feeding through duplicator are achieved. Here, there are 12 controllable factors and also an interaction between two factors affecting the feeding process. These factors and their levels are tabulated in Table 3.32:

Based on the L_{16} orthogonal array, for these factors, 16 different combinations are obtained, and each design is tested for 4 times. In the test phase, the numbers of sheets fed to the duplicator are counted. According to these counts, the categories are specified. The ranges for classes are tabulated in Table 3.33:

Table 3.32. Controllable Factors and Their Levels for the 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

Table 3.33. Range of the Number of Sheets for Each Class for the Duplicator Case

Classes	Range of Number of Sheets
I	Paper Feeding Failed
II	[1, 168]
III	[169, 336]
IV	[337, ∞]

As seen in the table, Class I indicates the situation where no paper sheets are successfully fed through the duplicator; Class IV indicates that 337 or more paper sheets successfully pass through the duplicator. It is clear that our class of interest is Class IV, i.e. we try to achieve the conditions such that more than or equal to 336 paper sheets are successfully fed through the machine. So, it is a larger-the-better type problem.

Test results that are generated by testing each of these 16 different parameter settings for 4 times are classified according to the count of successfully fed paper sheets. Numbers of observations for each class for the 16 experiments are tabulated in Table 3.34:

Since there is no missing value in the data, no manipulation is done for that purpose.

Table 3.34. Experimental Design and Class of Response for the Duplicator Case

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
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: Growing random forest trees and handling imbalanced data

The data is designed so as to run each of these 16 different factor level combinations for 4 times, so we ultimately have 64 observations in our data set. This data set is analyzed with Random Forest algorithm by using R-studio software (R Core Team, 2017).

Firstly, the numbers of attributes to use for each split are determined. As mentioned in Section 2.2.1, to decide this number, the square root of the number of factors affecting the response is taken, and the first integer value of this number is considered to be the best split value. So, for this example, it was equal to $\lfloor \sqrt{12} \rfloor = 3$. But, according to the R function, the best split value based on OOB error rate is obtained as 1. When the data is run with both these split values, it is seen that if the split value is 1, the OOB error estimate is lower. In fact, in his later study, Breiman (2001) specified that the best split value can be the first number of half of the square root of the number of factors affecting the response ($\lfloor \sqrt{p} / 2 \rfloor$) according to the data set. Based on this assumption, in our example, the best split value is equal to 1 $\left(\lfloor \sqrt{12} / 2 \rfloor \right)$. The OOB error rate based best split graph in R is shown in Figure 3.23.

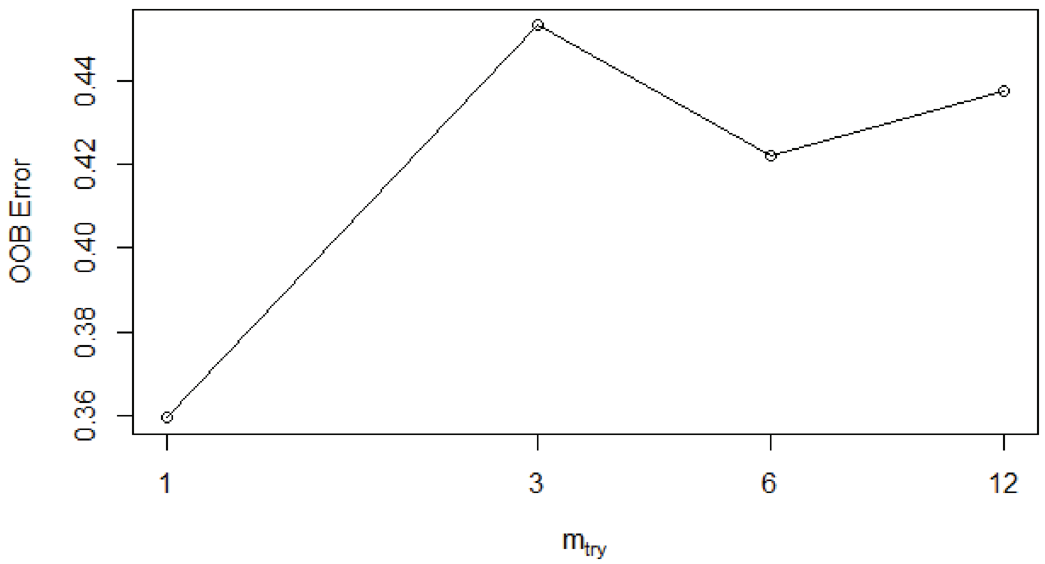


Figure 3.23. Best Split Value versus OOB Error Rate for the Duplicator Case

Moreover, the best number of trees in the forest is tested with different values between 100-10000, and the best value is set as 1000 according to OOB error rate and consuming of time.
Our data set is not balanced, i.e. trials in each class are not equal, not even close to each other. As seen in the Figure 3.24., while Class II constitutes 55% of the data; our class of interest, Class IV, only accounts for 22% of the data set (There are just 14 observations). The other classes have lower trials. There are 11 experiments for Class I (17%) and only 4 experiments for Class III (6%).

Since all classes do not have the same number of trials, we can say that this data set is imbalanced. By giving weights to classes, Random Forest can overcome this situation. To see the differences, we analyze the data both with unweighted Random Forest (each class has equal weight) and Random Forests with different weighted strategies as mentioned in Case Studies I and II.

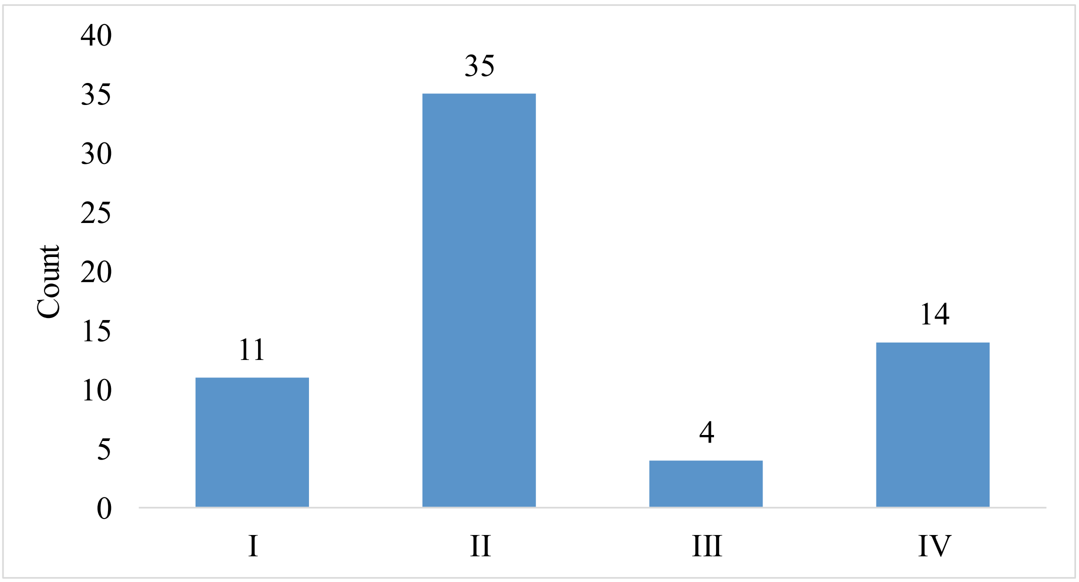


Figure 3.24. Distribution of Classes in the Data for the Duplicator Case

In this example, our problem is the larger-the-better type, so our class of interest is Class IV. We seek the parameter settings that provide Class IV. We try to achieve the best parameter settings for our class of interest, Class IV, by giving more cost to the misclassification of Class IV, and in this way, we could obtain the conditions which predict Class IV the best way possible. Best of all, Random Forest does it automatically, without any extra processing.

Weights of classes and their abbreviations are given in Table 3.35. Weights are given equally, linearly, piecewisely linear and exponentially, respectively. The weights are actually same as the weights in the Case Study-II (Inkjet Printer Example), because both have the same number of classes and both are Larger-the-Better type of problem.

Table 3.35. Class Weights of the Random Forests for the Duplicator Case

Weighting Strategies	Random Forests	Weights of Classes			
		I	II	III	IV
Equal Weights	RF_E	0.25	0.25	0.25	0.25
Linear Weights	RF_L	0.1	0.2	0.3	0.4
Piecewise Linear Weights	RF_P	0.166	0.167	0.167	0.5
Exponential Weights	RF_X	0.03	0.09	0.24	0.64

According to these weights, OOB error rate graphs of Random Forests formed with 1000 trees and with the split value equal to 1 are as given in Figures 3.25-3.28.

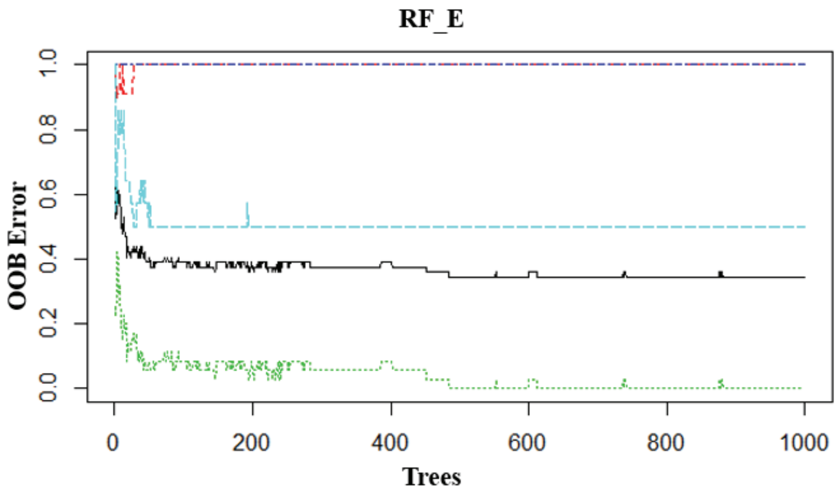


Figure 3.25. OOB Error Rate Graph of RF_E for the Duplicator Case

According to this graph, the black line refers to the OOB error estimate of the Random Forest with 34.38% on the average. Furthermore, red and blue lines correspond to Class I and Class III with average OOB error rates of 100%, respectively; the cyan line refers to Class IV with 50% OOB error rate on the average. Lastly, the green line refers to Class II with an average OOB error rate of 0%. As it is seen, the OOB error rate of Class II is very low, but our class of interest, Class IV, has a quite high OOB error rate. To avoid this case, we give more cost to the misclassification of Class IV (by giving more weight to Class IV than other classes). The OOB error rate graph of the first weighted Random Forest (linearly-weighted RF_L) is as in Figure 3.26.

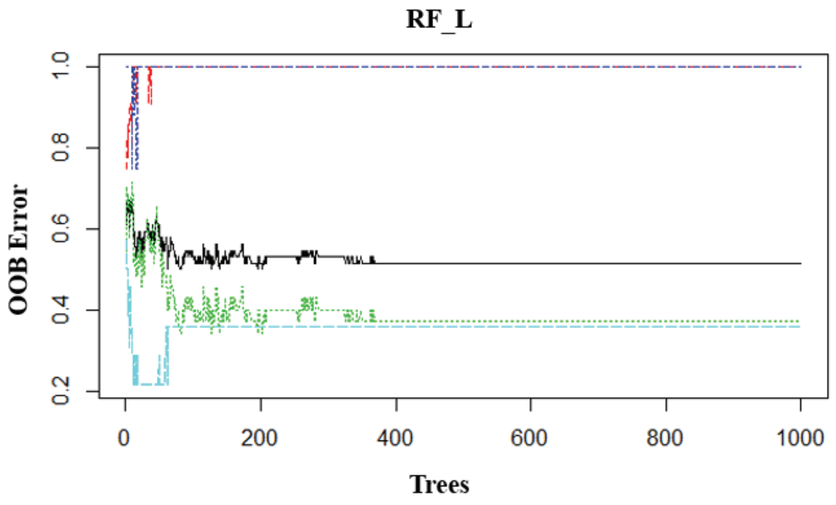


Figure 3.26. OOB Error Rate Graph of RF_L for the Duplicator Case

Average OOB error estimate of the Random Forest increased compared to that of RF_E. Average OOB error estimate of Random Forest has changed from 34.38% to 51.56%. And, here again, Class I and Class III have each an average 100% OOB error rate (red and blue line). While the average OOB error rate of Class II has increased (from 0% to 37% - green line), the average OOB error rate of Class IV has decreased (from 50% to 36% - cyan line). So, we conclude that giving linearly increasing weights does not affect the correctly classification of Class I and Class III, but affect those of Class II and Class IV. By giving more weight to Class IV, we have decreased the average OOB error rate of our class of interest.

The OOB error rate graph of the Random Forest created with piecewise linearly weights, RF_P, is as shown in Figure 3.27.

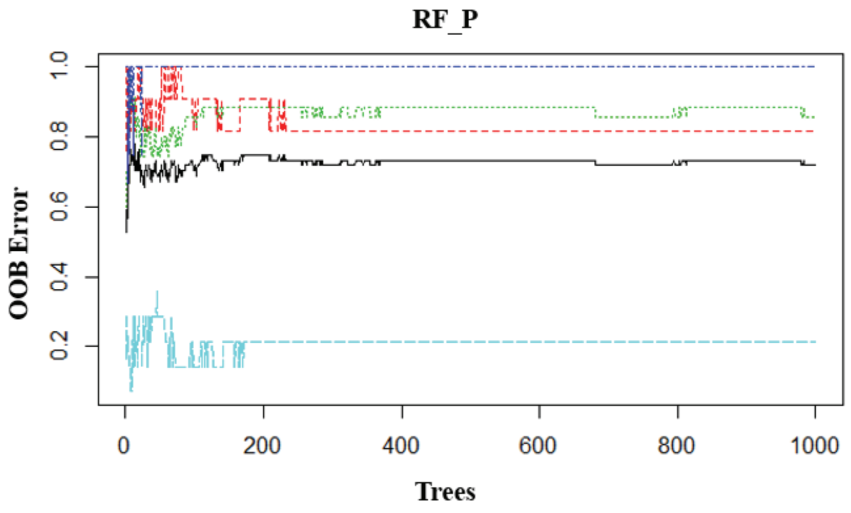


Figure 3.27. OOB Error Rate Graph of RF_P for the Duplicator Case

Here, we give half of the weights to the Class IV, and the remaining half of the weights are distributed among other classes equally. This Random Forest has the maximum average OOB error estimate (71.88% - Black Line) among the constructed Random Forests. This is the first time the average OOB error rate of Class I has ever decreased (from 100% to 82% - Red Line), but it is still very high. The average OOB error rate of Class II is at its maximum level (86% - Green Line). The average OOB error rate of Class III has not changed (still 100%). Despite all these, the average OOB error rate of Class IV is at its minimum level (21% - Cyan Line). So, we can say that this Random Forest can make satisfactory predictions for Class IV, but not for other classes.

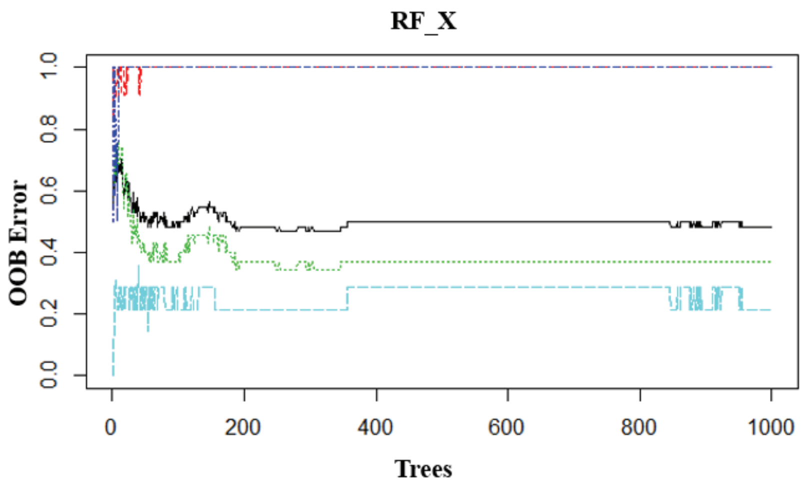


Figure 3.28. OOB Error Rate Graph of RF_X for the Duplicator Case

Lastly, we analyze the result of the Random Forest with exponential weights (RF_X) Average OOB error estimate of the Random Forest is 48.44% (black line). Its performance is better than both RF_L and RF_P. The average OOB error rate of Class IV is 21% (Cyan Line), and it is equal to the Class I OOB rate of RF_P. The average OOB error rate of Class II in this Random Forest is also lower (37% - Green Line) than those of RF_L and RF_P. And the OOB error rate of Class I and Class III are 100% (red and blue lines).

Shortly, when we evaluate the OOB error rate graphs of these four Random Forests, we observe that RF_E has the minimum average OOB error estimate (34.38%). The Random Forest with the second minimum average OOB error estimate is RF_X. RF_P is the Random Forest whose average OOB error estimate is maximum. The average OOB error rate of our class of interest (Class IV) is minimum in RF_P and RF_X (21%), but the general performance of the Random Forest is better in RF_X than RF_P. Since the number of observations of Class I and Class III is small, it is hard to analyze these classes for Random Forest and also for the other methods. Therefore, the class error rates of Class I and Class III are always high. On the other hand, the number of observations for Class II is the largest, and its average OOB error rate is generally low. So, we conclude that if we want to decide on a weighting strategy, we can choose exponentially weighting Random Forest (RF_X), because the general performance of RF_X can be sufficient, and, in addition, the prediction performance of our class of interest is the best in RF_X. If we cannot decide at this stage, we analyze classification performance of these Random Forests and then we can make a decision. Alternatively, we have to proceed to the optimization step.

To evaluate the performance measures of Random Forests, we use the performance measures mentioned in Section 2.2.1.6. Also, we compare these RFs with Logistic Regression by using these performance measures. Firstly, we examine the confusion matrices of RFs and Logistic Regression in Tables 3.36-3.40.

Table 3.36. Confusion Matrix of RF_E for the Duplicator Case

		Predicted				
	Classes	I	II	III	IV	Sum
Actual	I	0	10	0	1	11
	II	0	35	0	0	35
	III	0	4	0	0	4
	IV	0	7	0	7	14
	Sum	0	56	0	8	64

Table 3.37. Confusion Matrix of RF_L for the Duplicator Case

		Predicted				
	Classes	I	II	III	IV	Sum
Actual	I	2	6	2	1	11
	II	2	22	6	5	35
	III	0	0	4	0	4
	IV	0	0	4	10	14
	Sum	4	28	16	16	64

Table 3.38. Confusion Matrix of RF_P for the Duplicator Case

		Predicted				
	Classes	I	II	III	IV	Sum
Actual	I	6	2	1	2	11
	II	14	10	2	9	35
	III	0	0	1	3	4
	IV	0	0	0	14	14
	Sum	20	12	4	28	64

Table 3.39. Confusion Matrix of RF_X for the Duplicator Case

		Predicted				
	Classes	I	II	III		Sum
Actual	I	0	8	1	2	11
	II	0	24	2	9	35
	III	0	0	1	3	4
	IV	0	0	0	14	14
	Sum	0	32	4	28	64

Table 3.40. Confusion Matrix of Logistic Regression for the Duplicator Case

		Predicted				
	Classes	I	II	III	IV	Sum
Actual	I	0	10	0	1	11
	II	0	32	0	3	35
	III	0	3	0	1	4
	IV	0	3	0	11	14
	Sum	0	48	0	16	64

Here again, we create confusion matrices by testing all data sets, not the test set, as was the case for the previous examples. Measures which were calculated based on these confusion matrices are tabulated in Table 3.41.

As it is understood from Table 3.41, accuracies of RF_E and Logistic Regression are equal to each other and these are better than the accuracies of others. Since the data is imbalanced, it will be more accurate to use balanced accuracy. Balanced accuracies of all weighted Random Forests are better than the regular Random Forest and Logistic

Regression. For AUC, the same conclusion is valid. Moreover, Kappa and Balanced Kappa of weighted Random Forests are better than those corresponding to the regular Random Forest and generally Logistic Regression. For F-Score and G-Mean, results are changeable.

In addition to these comparisons, class based measures can also be compared. When we examine Table 3.42, for our class of interest (Class IV), recall is at the highest level in RF_P and RF_X, while precision and specificity are at the highest levels in RF_E. For the other classes, performances of the methods vary, so we could not generalize. Therefore, we can say that giving weights to the classes change the performance of the Random Forests. Evaluating all these performance measures, we concluded that the best performing method varies depending on each performance measure. It may be misleading to choose the best method in this stage. But, these performance measures can also be used as an indicator in the optimization step while deciding on the method to be selected.

Table 3.41. Comparison of Random Forests and Logistic Regression According to Performance Measures for the Duplicator Case

Random Forests	**Acc**	**Balanced Acc**	**Kappa**	**Balanced Kappa**	**F-Score**	**G-Mean**	**AUC**
RF_E	66%	38%	30%	-27%	45%	0%	60%
RF_L	59%	73%	40%	60%	48%	68%	80%
RF_P	48%	68%	31%	57%	37%	61%	78%
RF_X	61%	76%	38%	62%	45%	0%	77%
LR	66%	40%	33%	-17%	45%	0%	57%

Table 3.42. Comparison of Random Forest and Logistic Regression According to Class-Based Performance Measures for the Duplicator Example

Classe	**Recall**					**Precision**					**Specificity**				
s	RF_E	RF_L	RF_P	RF_X	LR	RF_E	RF_L	RF_P	RF_X	LR	RF_E	RF_L	RF_P	RF_X	LR
I	0%	18%	55%	0%	0%	0%	50%	30%	0%	0%	100%	96%	74%	100%	100%
II	100%	63%	29%	69%	94%	63%	79%	83%	75%	64%	28%	79%	43%	72%	35%
III	0%	100%	25%	25%	0%	0%	25%	25%	25%	0%	100%	80%	25%	95%	100%
IV	50%	71%	100%	100%	64%	88%	63%	50%	50%	75%	98%	88%	72%	72%	94%

When we analyze the importance of attributes, although the order of the importance of variables changed for each Random Forest, B is the most important attribute for each Random Forest for both measures (Mean Decrease Accuracy and Gini Index). Moreover, we can say I, F, K and D are the other significant factors. On the other hand, generally the least important variable is H. All variable importance plots are given in Appendix E.1. We can conclude that in cases where some factors have to be chosen for the trials, these important factors can be selected firstly.

Step 3: Estimating the probability of each class, and calculating expected value and variance for each experiment

Probabilities obtained from the different weighted Random Forests and expected value, variance and SNR values that are calculated based on these probabilities are given in Appendix E.2. For calculating the SNR, Equation (2.6) is used (since the problem is larger-the-better type).

For all Random Forests, experiment with the best design is the 14th experiment, $A_1B_1C_0D_0E_1F_1G_1H_0FxI_0I_1J_0K_0L_1$. In this parameter design, expected values are the closest to the desired value (4); 3.463, 3.915, 3.951 and 3.965 for RF_E, RF_L, RF_P, and RF_X, respectively. This experiment was previously tested for 4 times, and for all times, the experiment was already classified as Class IV. Also, this experiment has the maximum SNR values and the minimum variance (except RF_E) for all Random Forests.

Step 4: Find the optimal product/process design parameter settings for desired mean and minimum variance

In this problem, we want to achieve the best condition for the paper feeding phase of a duplicator machine at high speed. In other words, our aim is to find the parameter settings such that the maximum number of paper sheets successfully passes through the duplicator. So, these parameter settings should have the maximum estimated probability of Class IV, and also, the expected value of these parameter settings are to be close or equal to 4, and the variances of them are to be minimum. To solve the problem, as mentioned before in Section 3.1, the ε-Constraint and the SNR methods are applied.

a) Obtain the optimal parameter settings by using ε-Constraint method

To develop the mathematical model needed by this method, firstly, the expected value and the variance regression models are generated by using Minitab 18 (2018). For the regression models, both main and interaction effects of factors are considered. For this purpose, we perform firstly stepwise regression with both main and interaction of factor. After that, by adding the most significant factors and interactions or removing the least significant factors and interactions from the model we obtain more robust regression models. For the regression, all factors have only two levels, so we set all of them as categorical. For all different Random Forests, weighted schemes, we obtain different expected value and variance models. The expected value and the variance models created for these Random Forests are given in Figure 3.29-32.

EV=2.02737+0.0B_0+0.54810B_1+0.0E_0-0.12819E_1+0.0G_0
+0.07117G_1+0.0A*B_00+0.0A*B_01+0.0A*B_10+0.00833A*B_11
+0.0A*L_00+0.0A*L_01+0.0A*L_10+0.14754A*L_11+0.0B*E_00
+0.0B*E_01+0.0B*E_10+0.24371B*E_11+0.0C*FxI_00+0.0C*FxI_01
+0.0C*FxI_10-0.24379C*FxI_11+0.0D*G_00+0.0D*G_01+0.0D*G_10
-0.47225D*G_11+0.0E*K_00+0.0E*K_01+0.0E*K_10+0.14533E*K_11
+0.0F*G_00+0.0F*G_01+0.0F*G_10+0.54517F*G_11+0.0H*I_00
+0.0H*I_01 +0.0H*I_10-0.03608H*I_11+0.0H*J_00+0.0H*J_01+0.0H*J_10
-0.04442H*J_11+0.0H*K_00+0.0H*K_01+0.0H*K_10+0.00854H*K_11

R^2=100.00% R^2(adj)=100.00% R^2(pred)=99.98%

Var= 0.28921+0.0B_0+0.6102B_1+0.0C_0+0.05340C_1+0.0E_0-0.06116E_1
+0.0FxI_0-0.08130FxI_1+0.0K_0+0.11857K_1+0.0B*C_00+0.0B*C_01
+0.0B*C_10+0.0676B*C_11+0.0B*H_00+0.0B*H_01+0.0B*H_10
+0.22470B*H_11+0.0B*K_00+0.0B*K_01+0.0B*K_10-0.4791B*K_11
+0.0C*D_00+0.0C*D_01+0.0C*D_10-0.08861C*D_11+0.0D*H_00
+0.0D*H_01+0.0D*H_10-0.09360D*H_11+0.0E*FxI_00+0.0E*FxI_01
+0.0E*FxI_10+0.2534E*FxI_11+0.0F*H_00+0.0F*H_01+0.0F*H_10
+0.03479F*H_11

R^2=99.97% R^2(adj)=99.87% R^2(pred)=99.30%

Figure 3.29. Expected Value and Variance Regression Equations of RF_E for the Duplicator Case

EV = 2.39004+0.0B_0+1.01968B_1+0.0A*E_00+0.0A*E_01+0.0A*E_10
+0.04724A*E_11+0.0B*D_00+0.0B*D_01+0.0B*D_10-0.46235B*D_11
+0.0B*I_00+0.0B*I_01+0.0B*I_10+0.39848B*I_11+0.0C*D_00+0.0C*D_01
+0.0C*D_10+0.04944C*D_11+0.0C*FxI_00+0.0C*FxI_01+0.0C*FxI_10
-0.19368C*FxI_11+0.0D*H_00+0.0D*H_01+0.0D*H_10+0.10692D*H_11
+0.0D*K_00+0.0D*K_01+0.0D*K_10-0.10452D*K_11+0.0D*L_00
+0.0D*L_01+0.0D*L_10+0.17110D*L_11+0.0E*J_00+0.0E*J_01+0.0E*J_10
-0.02012E*J_11+0.0G*L_00+0.0G*L_01+0.0G*L_10+0.05994G*L_11
+0.0H*I_00+0.0H*I_01+0.0H*I_10-0.00749H*I_11

R^2=100.00% R^2(adj)=100.00% R^2(pred)=99.96%

Var = 0.9738+0.0B_0-0.3348B_1+0.0E_0+0.1280E_1+0.0F_0+0.0747F_1
+0.0A*L_00+0.0A*L_01+0.0A*L_10-.1251A*L_11+0.0B*E_00+0.0B*E_01
+0.0B*E_10-0.3745B*E_11+0.0C*FxI_00+0.0C*FxI_01+0.0C*FxI_10
+0.1462C*FxI_11+0.0F*G_00+0.0F*G_01+0.0F*G_10-0.2421F*G_11
+0.0F*J_00+0.0F*J_01+0.0F*J_10-0.1575F*J_11

R^2=99.64% R^2(adj)=99.23% R^2(pred)=97.90%

Figure 3.30. Expected Value and Variance Regression Equations of RF_L for the Duplicator Case

EV=2.21039+0.0B_0+1.14622B_1+0.0G_0-0.3360G_1+0.0A*C_00
+0.0A*C_01+0.0A*C_10-0.06480A*C_11+0.0A*D_00+0.0A*D_01
+0.0A*D_10+0.02067A*D_11+0.0A*F_00+0.0A*F_01+0.0A*F_10
+0.14443A*F_11+0.0D*G_00+0.0D*G_01+0.0D*G_10-0.28473D*G_11
+0.0D*H_00+0.0D*H_01+0.0D*H_10-0.03773D*H_11+0.0E*J_00
+0.0E*J_01+0.0E*J_10-0.19155E*J_11+0.0E*K_00+0.0E*K_01+0.0E*K_10
+0.27550E*K_11+0.0F*G_00+0.0F*G_01+0.0F*G_10+0.78802F*G_11
+0.0H*L_00+0.0H*L_01+0.0H*L_10+0.35347H*L_11+0.0FxI*J_00
+0.0FxI*J_01+0.0FxI*J_10+0.03955FxI*J_11

R^2=100.00% R^2(adj)=99.98% R^2(pred)=99.92%

Var= 1.34133+0.0B_0-0.46810B_1+0.0E_0-0.04494E_1+0.0F_0+0.03145F_1
+0.0G_0+0.37407G_1+0.0H_0+0.10554H_1+0.0L_0+0.02674L_1
+0.0A*B_00+0.0A*B_01+0.0A*B_10-0.20476A*B_11+0.0B*H_00
+0.0B*H_01+0.0B*H_10-0.36689B*H_11+0.0C*J_00+0.0C*J_01
+0.0C*J_10-0.03134C*J_11+0.0D*FxI_00+0.0D*FxI_01+0.0D*FxI_10
+0.11262D*FxI_11+0.0F*G_00+0.0F*G_01+0.0F*G_10-0.54187F*G_11
+0.0G*L_00+0.0G*L_01+0.0G*L_10-0.39874G*L_11+0.0I*K_00
+0.0I*K_01 +0.0I*K_10-0.02810I*K_11

R^2=100.00% R^2(adj)=100.00% R^2(pred)=99.80%

Figure 3.31. Expected Value and Variance Regression Equations of RF_P for the Duplicator Case

EV=2.54290+0.0A_0+0.01012A_1+0.0B_0+0.40558B_1+0.0F_0
+0.04993F_1+0.0I_0-0.05625I_1+0.0B*E_00+0.0B*E_01+0.0B*E_10
+0.12370B*E_11+0.0B*F_00+0.0B*F_01+0.0B*F_10+0.31725B*F_11
+0.0B*H_00+0.0B*H_01+0.0B*H_10+0.19267B*H_11+0.0B*I_00
+0.0B*I_01+0.0B*I_10+0.46075B*I_11+0.0B*L_00
+0.0B*L_01+0.0B*L_10+0.15172B*L_11+0.0C*E_00+0.0C*E_01
+0.0C*E_10-0.05730C*E_11+0.0E*G_00+0.0E*G_01+0.0E*G_10
-0.06538E*G_11+0.0E*L_00+0.0E*L_01+0.0E*L_10+0.02428E*L_11
+0.0H*L_00+0.0H*L_01+0.0H*L_10 +0.07567H*L_11

R^2=100.00% R^2(adj)=100.00% R^2(pred)=99.89%

Var=1.01234+0.0B_0-0.690055B_1+0.0F_0+0.27677F_1+0.0G_0
+0.32809G_1+0.0A*E_00+0.0A*E_01+0.0A*E_10-0.096087A*E_11
+0.0B*FxI_00+0.0B*FxI_01+0.0B*FxI_10+0.11807B*FxI_11
+0.0D*I_00+0.0D*I_01+0.0D*I_10+0.06336D*I_11+0.0E*F_00+0.0E*F_01
+0.0E*F_10+0.01085E*F_11+0.0F*G_00+0.0F*G_01+0.0F*G_10
-0.62482F*G_11+0.0F*J_00+0.0F*J_01+0.0F*J_10-0.207278F*J_11
+0.0H*I_00+0.0H*I_01+0.0H*I_10-0.00822H*I_11+0.0H*J_00+0.0H*J_01
+0.0H*J_10+0.09425H*J_11+0.0H*K_00+0.0H*K_01+0.0H*K_10
-0.238980H*K_11+0.0I*L_00+0.0L_01+0.0I*L_10-0.151301I*L_11

R^2=100.00% R^2(adj)=100.00% R^2(pred)=99.99%

Figure 3.32. Expected Value and Variance Regression Equations of RF_X for the Duplicator Case

To determine the best levels of parameters, for the desired expected value, 4, and the minimum variance, these regression models are used to formulate the RPD problem as a Multi Objective Optimization problem using the ε-Constraint Method as mentioned in Section 2.1.2. The models for each Random Forest are as follows:

1. **For RF_E:**

$$\text{Min } (\hat{V}) + (|\hat{E} - 4| \times 10^{-4})$$

s.t.

$$|\hat{E} - 4| \leq \varepsilon$$

$$\hat{E} \geq 0$$

$$\hat{V} \geq 0$$

$$\hat{E} = 2.02737 + 0.0 \times B_0 + 0.5481 \times B_1 + 0.0 \times E_0 - 0.12819 \times E_1 + 0.0 \times G_0 + 0.07117 \times G_1 + 0.0 \times A_0 \times B_0 + 0.0 \times A_0 \times B_1 + 0.0 \times A_1 \times B_0 + 0.00833 \times A_1 \times B_1 + 0.0 \times A_0 \times L_0 + 0.0 \times A_0 \times L_1 + 0.0 \times A_1 \times L_0 + 0.14754 \times A_1 \times L_1 + 0.0 \times B_0 \times E_0 + 0.0 \times B_0 \times E_1 + 0.0 \times B_1 \times E_0 + 0.24371 \times B_1 \times E_1 + 0.0 \times C_0 \times FxI_0 + 0.0 \times C_0 \times FxI_1 + 0.0 \times C_1 \times FxI_0 - 0.24379 \times C_1 \times FxI_1 + 0.0 \times D_0 \times G_0 + 0.0 \times D_0 \times G_1 + 0.0 \times D_1 \times G_0 - 0.47225 \times D_1 \times G_1 + 0.0 \times E_0 \times K_0 + 0.0 \times E_0 \times K_1 + 0.0 \times E_1 \times K_0 + 0.14533 \times E_1 \times K_1 + 0.0 \times F_0 \times G_0 + 0.0 \times F_0 \times G_1 + 0.0 \times F_1 \times G_0 + 0.54517 \times F_1 \times G_1 + 0.0 \times H_0 \times I_0 + 0.0 \times H_0 \times I_1 + 0.0 \times H_1 \times I_0 - 0.03608 \times H_1 \times I_1 + 0.0 \times H_0 \times J_0 + 0.0 \times H_0 \times J_1 + 0.0 \times H_1 \times J_0 - 0.04442 \times H_1 \times J_1 + 0.0 \times H_0 \times K_0 + 0.0 \times H_0 \times K_1 + 0.0 \times H_1 \times K_0 + 0.00854 \times H_1 \times K_1$$

$$\hat{V} = 0.28921 + 0.0 \times B_0 + 0.6102 \times B_1 + 0.0 \times C_0 + 0.0534 \times C_1 + 0.0 \times E_0 - 0.06116 \times E_1 + 0.0 \times FxI_0 - 0.0813 \times FxI_1 + 0.0 \times K_0 + 0.11857 \times K_1 + 0.0 \times B_0 \times C_0 + 0.0 \times B_0 \times C_1 + 0.0 \times B_1 \times C_0 + 0.0676 \times B_1 \times C_1 + 0.0 \times B_0 \times H_0 + 0.0 \times B_0 \times H_1 + 0.0 \times B_1 \times H_0 + 0.2247 \times B_1 \times H_1 + 0.0 \times B_0 \times K_0 + 0.0 \times B_0 \times K_1 + 0.0 \times B_1 \times K_0 - 0.4791 \times B_1 \times K_1 + 0.0 \times C_0 \times D_0 + 0.0 \times C_0 \times D_1 + 0.0 \times C_1 \times D_0 - 0.08861 \times C_1 \times D_1 + 0.0 \times D_0 \times H_0 + 0.0 \times D_0 \times H_1 + 0.0 \times D_1 \times H_0 - 0.0936 \times D_1 \times H_1 + 0.0 \times E_0 \times FxI_0 + 0.0 \times E_0 \times FxI_1 + 0.0 \times E_1 \times FxI_0 + 0.2534 \times E_1 \times FxI_1 + 0.0 \times F_0 \times H_0 + 0.0 \times F_0 \times H_1 + 0.0 \times F_1 \times H_0 + 0.03479 \times F_1 \times H_1$$

$A_0, A_1, B_0, B_1, C_0, C_1, D_0, D_1, E_0, E_1, F_0, F_1, G_0, G_1, H_0, H_1, FxI_0, FxI_1, I_0, I_1, J_0, J_1, K_0, K_1, L_0, L_1 \in \{0, 1\}$

2. For RF_L:

$$\text{Min } (\hat{V}) + (|\hat{E} - 4| \times 10^{-4})$$

s.t.

$$|\hat{E} - 4| \leq \varepsilon$$

$$\hat{E} \geq 0$$

$$\hat{V} \geq 0$$

$$\hat{E} = 2.39004 + 0.0\times B_0 + 1.01968\times B_1 + 0.0\times A_0\times E_0 + 0.0\times A_0\times E_1 + 0.0\times A_1\times E_0 + 0.04724\times A_1\times E_1 + 0.0\times B_0\times D_0 + 0.0\times B_0\times D_1 + 0.0\times B_1\times D_0 - 0.46235\times B_1\times D_1 + 0.0\times B_0\times I_0 + 0.0\times B_0\times I_1 + 0.0\times B_1\times I_0 + 0.39848\times B_1\times I_1 + 0.0\times C_0\times D_0 + 0.0\times C_0\times D_1 + 0.0\times C_1\times D_0 + 0.04944\times C_1\times D_1 + 0.0\times C_0\times FxI_0 + 0.0\times C_0\times FxI_1 + 0.0\times C_1\times FxI_0 - 0.19368\times C_1\times FxI_1 + 0.0\times D_0\times H_0 + 0.0\times D_0\times H_1 + 0.0\times D_1\times H_0 + 0.10692\times D_1\times H_1 + 0.0\times D_0\times K_0 + 0.0\times D_0\times K_1 + 0.0\times D_1\times K_0 - 0.10452\times D_1\times K_1 + 0.0\times D_0\times L_0 + 0.0\times D_0\times L_1 + 0.0\times D_1\times L_0 + 0.1711\times D_1\times L_1 + 0.0\times E_0\times J_0 + 0.0\times E_0\times J_1 + 0.0\times E_1\times J_0 - 0.02012\times E_1\times J_1 + 0.0\times G_0\times L_0 + 0.0\times G_0\times L_1 + 0.0\times G_1\times L_0 + 0.05994\times G_1\times L_1 + 0.0\times H_0\times I_0 + 0.0\times H_0\times I_1 + 0.0\times H_1\times I_0 - 0.00749\times H_1\times I_1$$

$$\hat{V} = 0.9738 + 0.0\times B_0 - 0.3348\times B_1 + 0.0\times E_0 + 0.128\times E_1 + 0.0\times F_0 + 0.0747\times F_1 + 0.0\times A_0\times L_0 + 0.0\times A_0\times L_1 + 0.0\times A_1\times L_0 - 0.1251\times A_1\times L_1 + 0.0\times B_0\times E_0 + 0.0\times B_0\times E_1 + 0.0\times B_1\times E_0 - 0.3745\times B_1\times E_1 + 0.0\times C_0\times FxI_0 + 0.0\times C_0\times FxI_1 + 0.0\times C_1\times FxI_0 + 0.1462\times C_1\times FxI_1 + 0.0\times F_0\times G_0 + 0.0\times F_0\times G_1 + 0.0\times F_1\times G_0 - 0.2421\times F_1\times G_1 + 0.0\times F_0\times J_0 + 0.0\times F_0\times J_1 + 0.0\times F_1\times J_0 - 0.1575\times F_1\times J_1$$

$A_0, A_1, B_0, B_1, C_0, C_1, D_0, D_1, E_0, E_1, F_0, F_1, G_0, G_1, H_0, H_1, FxI_0, FxI_1, I_0, I_1, J_0, J_1, K_0, K_1, L_0, L_1 \in \{0, 1\}$

3. For RF_P:

$$\text{Min } (\hat{V}) + (|\hat{E} - 4| \times 10^{-4})$$

s.t.

$$|\hat{E} - 4| \leq \varepsilon$$

$$\hat{E} \geq 0$$

$$\hat{V} \geq 0$$

$$\hat{E} = 2.21039 + 0.0 \times B_0 + 1.14622 \times B_1 + 0.0 \times G_0 - 0.336 \times G_1 + 0.0 \times A_0 \times C_0 + 0.0 \times A_0 \times C_1 + 0.0 \times A_1 \times C_0 - 0.0648 \times A_1 \times C_1 + 0.0 \times A_0 \times D_0 + 0.0 \times A_0 \times D_1 + 0.0 \times A_1 \times D_0 + 0.02067 \times A_1 \times D_1 + 0.0 \times A_0 \times F_0 + 0.0 \times A_0 \times F_1 + 0.0 \times A_1 \times F_0 + 0.14443 \times A_1 \times F_1 + 0.0 \times D_0 \times G_0 + 0.0 \times D_0 \times G_1 + 0.0 \times D_1 \times G_0 - 0.28473 \times D_1 \times G_1 + 0.0 \times D_0 \times H_0 + 0.0 \times D_0 \times H_1 + 0.0 \times D_1 \times H_0 - 0.03773 \times D_1 \times H_1 + 0.0 \times E_0 \times J_0 + 0.0 \times E_0 \times J_1 + 0.0 \times E_1 \times J_0 - 0.19155 \times E_1 \times J_1 + 0.0 \times E_0 \times K_0 + 0.0 \times E_0 \times K_1 + 0.0 \times E_1 \times K_0 + 0.2755 \times E_1 \times K_1 + 0.0 \times F_0 \times G_0 + 0.0 \times F_0 \times G_1 + 0.0 \times F_1 \times G_0 + 0.78802 \times F_1 \times G_1 + 0.0 \times H_0 \times L_0 + 0.0 \times H_0 \times L_1 + 0.0 \times H_1 \times L_0 + 0.35347 \times H_1 \times L_1 + 0.0 \times FxI_0 \times J_0 + 0.0 \times FxI_0 \times J_1 + 0.0 \times FxI_1 \times J_0 + 0.03955 \times FxI_1 \times J_1$$

$$\hat{V} = 1.34133 + 0.0 \times B_0 - 0.4681 \times B_1 + 0.0 \times E_0 - 0.04494 \times E_1 + 0.0 \times F_0 + 0.03145 \times F_1 + 0.0 \times G_0 + 0.37407 \times G_1 + 0.0 \times H_0 + 0.10554 \times H_1 + 0.0 L_0 + 0.02674 \times L_1 + 0.0 \times A_0 \times B_0 + 0.0 \times A_0 \times B_1 + 0.0 \times A_1 \times B_0 - 0.20476 \times A_1 \times B_1 + 0.0 \times B_0 \times H_0 + 0.0 \times B_0 \times H_1 + 0.0 \times B_1 \times H_0 - 0.36689 \times B_1 \times H_1 + 0.0 \times C_0 \times J_0 + 0.0 \times C_0 \times J_1 + 0.0 \times C_1 \times J_0 - 0.03134 \times C_1 \times J_1 + 0.0 \times D_0 \times FxI_0 + 0.0 \times D_0 \times FxI_1 + 0.0 \times D_1 \times FxI_0 + 0.11262 \times D_1 \times FxI_1 + 0.0 \times F_0 \times G_0 + 0.0 \times F_0 \times G_1 + 0.0 \times F_1 \times G_0 - 0.54187 \times F_1 \times G_1 + 0.0 \times G_0 \times L_0 + 0.0 \times G_0 \times L_1 + 0.0 \times G_1 \times L_0 - 0.39874 \times G_1 \times L_1 + 0.0 \times I_0 \times K_0 + 0.0 \times I_0 \times K_1 + 0.0 \times I_1 \times K_0 - 0.0281 \times I_1 \times K_1$$

$A_0, A_1, B_0, B_1, C_0, C_1, D_0, D_1, E_0, E_1, F_0, F_1, G_0, G_1, H_0, H_1, FxI_0, FxI_1, I_0, I_1, J_0, J_1, K_0, K_1, L_0, L_1 \in \{0, 1\}$

4. For RF_X:

$$\text{Min } (\hat{V}) + (|\hat{E} - 4| \times 10^{-4})$$

s.t.

$$|\hat{E} - 4| \leq \varepsilon$$

$$\hat{E} \geq 0$$

$$\hat{V} \geq 0$$

$\hat{E} = 2.54290 + 0.0 \times A_0 + 0.01012 \times A_1 + 0.0 \times B_0 + 0.40558 \times B_1 + 0.0 \times F_0 + 0.04993 \times F_1 + 0.0 \times I_0 - 0.05625 \times I_1 + 0.0 \times B_0 \times E_0 + 0.0 \times B_0 \times E_1 + 0.0 \times B_1 \times E_0 + 0.1237 \times B_1 \times E_1 + 0.0 \times B_0 \times F_0 + 0.0 \times B_0 \times F_1 + 0.0 \times B_1 \times F_0 + 0.31725 \times B_1 \times F_1 + 0.0 \times B_0 \times H_0 + 0.0 \times B_0 \times H_1 + 0.0 \times B_1 \times H_0 + 0.19267 \times B_1 \times H_1 + 0.0 \times B_0 \times I_0 + 0.0 \times B_0 \times I_1 + 0.0 \times B_1 \times I_0 + 0.46075 \times B_1 \times I_1 + 0.0 \times B_0 \times L_0 + 0.0 \times B_0 \times L_1 + 0.0 \times B_1 \times L_0 + 0.15172 \times B_1 \times L_1 + 0.0 \times C_0 \times E_0 + 0.0 \times C_0 \times E_1 + 0.0 \times C_1 \times E_0 - 0.0573 \times C_1 \times E_1 + 0.0 \times E_0 \times G_0 + 0.0 \times E_0 \times G_1 + 0.0 \times E_1 \times G_0 - 0.06538 \times E_1 \times G_1 + 0.0 \times E_0 \times L_0 + 0.0 \times E_0 \times L_1 + 0.0 \times E_1 \times L_0 + 0.02428 \times E_1 \times L_1 + 0.0 \times H_0 \times L_0 + 0.0 \times H_0 \times L_1 + 0.0 \times H_1 \times L_0 + 0.07567 \times H_1 \times L_1$

$\hat{V} = 1.01234 + 0.0 \times B_0 - 0.690055 \times B_1 + 0.0 \times F_0 + 0.27677 \times F_1 + 0.0 \times G_0 + 0.32809 \times G_1 + 0.0 \times A_0 \times E_0 + 0.0 \times A_0 \times E_1 + 0.0 \times A_1 \times E_0 - 0.096087 \times A_1 \times E_1 + 0.0 \times B_0 \times FxI_0 + 0.0 \times B_0 \times FxI_1 + 0.0 \times B_1 \times FxI_0 + 0.11807 \times B_1 \times FxI_1 + 0.0 \times D_0 \times I_0 + 0.0 \times D_0 \times I_1 + 0.0 \times D_1 \times I_0 + 0.06336 \times D_1 \times I_1 + 0.0 \times E_0 \times F_0 + 0.0 \times E_0 \times F_1 + 0.0 \times E_1 \times F_0 + 0.01085 \times E_1 \times F_1 + 0.0 \times F_0 \times G_0 + 0.0 \times F_0 \times G_1 + 0.0 \times F_1 \times G_0 - 0.62482 \times F_1 \times G_1 + 0.0 \times F_0 \times J_0 + 0.0 \times F_0 \times J_1 + 0.0 \times F_1 \times J_0 - 0.207278 \times F_1 \times J_1 + 0.0 \times H_0 \times I_0 + 0.0 \times H_0 \times I_1 + 0.0 \times H_1 \times I_0 - 0.00822 \times H_1 \times I_1 + 0.0 \times H_0 \times J_0 + 0.0 \times H_0 \times J_1 + 0.0 \times H_1 \times J_0 + 0.09425 \times H_1 \times J_1 + 0.0 \times H_0 \times K_0 + 0.0 \times H_0 \times K_1 + 0.0 \times H_1 \times K_0 - 0.23898 \times H_1 \times K_1 + 0.0 \times I_0 \times L_0 + 0.0 \times I_0 \times L_1 + 0.0 \times I_1 \times L_0 - 0.151301 \times I_1 \times L_1$

$A_0, A_1, B_0, B_1, C_0, C_1, D_0, D_1, E_0, E_1, F_0, F_1, G_0, G_1, H_0, H_1, FxI_0, FxI_1, I_0, I_1, J_0, J_1, K_0, K_1, L_0, L_1 \in \{0, 1\}$

We have used MATLAB/BARON (2013) software to solve models based on different ε values. The optimal solutions according to different ε values are in Table 3.43.

We have an opportunity to select the best parameter setting suitable for our purposes. So, if our main aim is to have the expected value to be very close to the target value, we can choose the 5th or 7th solutions since the expected values, which are calculated by using their expected value regression model, under these solutions are 3.9926 and 3.9894, respectively. However, when the expected values of the 5th or 7th are calculated by using Equation (3.2) after predicting the probabilities from related Random Forests are 3.9430 and 3.3800, respectively. On the other hand, if our main purpose is to have the minimum variance, we can select 8th or 9th solutions since variance values, which is calculated with their regression model, of these solutions is lower than the others, they are 0.0163 and 0.0055 respectively. But, when we calculate the variance values of these solutions by using Equation (3.3) after predicting the probabilities from related Random Forests are 0.8693 and 0.8209, respectively. These analyses show that repeating the algorithm after collecting new data would increase the reliability of the model.

Table 3.43. Solutions of ε-Constraint Method for the Duplicator Case

Solutions		**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**FxI**	**I**	**J**	**K**	**L**	ε	$\hat{EV}$	$\hat{Var}$
1	RF_E	1	1	0	0	1	1	1	0	0	0	0	1	1	0.3915	3.6085	0.4777
2		1	1	0	0	0	1	1	0	1	0	0	1	1	0.6523	3.3477	0.4576
3	RF_L	1	1	0	0	1	1	1	0	0	1	0	0	1	0.0846	3.9154	0.1000
4		1	1	0	0	1	1	1	0	0	1	1	0	0	0.1647	3.8353	0.0676
5	RF_P	1	1	0	0	0	1	1	0	1	1	1	1	1	0.0074	3.9926	0.1320
6		0	1	1	0	1	1	1	1	0	1	1	0	1	0.0294	3.9706	0.0273
7	RF_X	1	1	1	0	1	1	0	1	0	1	1	1	0	0.0107	3.9894	0.1536
8		0	1	0	1	1	1	1	1	0	1	1	1	0	0.0289	3.9712	0.0163
9		1	1	0	1	0	1	1	1	0	1	1	1	0	0.0771	3.9230	0.0055

One of the points to pay attention to here is that the worst results for both the expected value and the variance are in the solutions of the equal-weighted Random Forest; that is, RF_E. As seen in Table 3.43, solution numbers 1 and 2 are the worst solutions in terms of both the variance and the expected value. Furthermore, some of weighted Random Forests results could obtain epsilon very close to 0, while the best result of epsilon in RF_E is equal to 0.3915. As it is seen on Table 3.43, we have achieved satisfactory results based on both the expected value and the variance in the solution of all weighted Random Forests. Therefore, it would be more logical to choose one of the weighted Random Forests. By considering both optimization and classification performance of forests, we can determine which forest has to be selected. RF_X shows a slightly better performance than other Random Forests. So, we can select the exponentially-weighted Random Forest; that is, RF_X, as the method. Alternatively, RF_P also has satisfactory results and can be considered as another preferable one. RF_L can be considered as another alternative because the predicted probability of Class IV, the expected value, and the variance of one of the best solution of it result in good way.

b) Obtain the optimal parameter settings by using SNR

We also solve the problem to achieve the best parameter design by using SNR method. For that purpose, ordinary least squares and response optimizer in Minitab 18 (2018) is used to fit the model. During fitting the SNR models stepwise regression is used for the starting point and for further iterations is generated based on p-values. Different starting points for response optimizer are also tried to achieve the best parameter settings. Since we have four different Random Forest models, we also have four different SNR solutions. According to these different Random Forest models, SNR-based solutions are given in Table 3.44.

As it is seen from the table, each Random Forest gives a different solution. Since the models have very high R-Sq, R-Sq(adj), and R-Sq(pred) values, we can say that these four models are highly adequate.

Table 3.44. Solution of the SNR model for the Duplicator Case

Random Forests	**Regression Results**			**Optimizer Results**													$\hat{SNR}$
	R-Sq	R-Sq (adj)	R-Sq (pred)	A	B	C	D	E	F	G	H	FxI	I	J	K	L	
RF_E	100%	100%	99.95%	0	1	-	-	0	1	1	-	1	1	0	0	1	10.7698
RF_L	100%	100%	99.94%	1	1	0	0	-	1	1	0	0	1	-	-	1	11.7663
RF_P	100%	100%	100%	1	1	0	0	-	1	1	0	-	1	0	-	1	11.8395
RF_X	100%	100%	99.99%	-	1	0	0	1	1	1	0	0	1	0	0	1	11.9070

The SNR values of the weighted Random Forests are better than that of the regular Random Forest, as we expected. It shows that giving weight to the classes is the important issue for the Random Forest method. Moreover, when we look at the classification performance of the models, OOB error estimate of Random Forest and also OOB error of Class IV (our class of interest) are better in RF_X than other weighted models. Therefore, it might be logical to choose the exponentially weighted Random Forest, RF_X. On the other hand, performance of RF_L and RF_P are also satisfactory.

As it is observed, there are some insignificant factors for the solutions. Factor C, D, and H are insignificant for RF_E; factors E, J and K for RF_L; factors E, FxI, and K for RF_P; factor A for RF_X. We can set any appropriate values to these factors, but generally, economic levels are preferred.

Two different optimization method, ε-Constraint and SNR methods are carried out to find the best parameter settings to have Class IV for the Duplicator Case. As we mentioned before, while in the SNR method, the expected value and variance are combined into a single objective function, SNR, in ε-Constraint method the expected value and variance are taken into account separately. One can choose any of these methods.

Step 5: Confirmation of results and revisiting the problem

When we examine results of both methods, it can be stated that most of their results are different from the tested ones while some of the others are same as the tested one.

Since we borrow the data from study of Logothetis and Wynn (1994), we do not have the chance to replicate the optimal solutions. For solutions that are different from the tested ones, different ways mentioned in Section 3.1 (Step 5) can be performed for confirmation. On the other hand, for the optimal solution that are same the tested one, Logothetis and Wynn (1994) have run the parameter design for 4 times, and they have obtained 4 times Class IV. Like Case Study-I and II, we apply Exact Multinomial Test to confirm our results with Logothetis and Wynn’s replications. During the test R-studio (R Core Team, 2017), which use Monte Carlo approach, is used. For aforementioned experimental design, we have found p-value as 1. This means that fail to reject the null hypothesis, that is there is no difference between our result and Logothetis and Wynn's replication. Moreover, we have observed that the reliability of the test increases when the number of experiments is increased.

Our problem is the Larger-the-Better type, so we are seeking the parameter designs that feed maximum number of sheets to a duplicator, and this case refers to Class IV. So, our aim is to find the parameter designs that give the highest probability of Class IV, and also, we would like to have an expected value that is as close as possible to 4, minimum variance and maximum SNR values. As we expected, Random Forests, whose weights of Class IV are higher, find the solution with higher SNR values. We can conclude that, while giving weights to the classes, especially giving more weight to our class of interest increases the prediction power of the class, since their SNR values are higher than those of the unweighted Random Forest. The results show that the estimated values of SNR are higher in weighted Random Forests than the regular Random Forest.

This example data set is also analyzed before by Karabulut (2013) by using Logistic Regression method, and she obtains the best parameter design as $B_1F_1K_0L_1$. In her solution estimates the probability of Class IV is 0.99, while the estimated expected value, variance and SNR value were 3.9841, 0.0765 and 11.9835, respectively.

In our solutions, we obtain the parameter designs whose expected value is equal to 3.9926 and the variance equals to 0.0055, which are two of the optimal solution of ε-Constraint method of RF_P and RF_X given in Table 3.43 with solution numbers 5 and 9 ($A_1B_1C_0D_0E_0F_1G_1H_0FxI_1I_1J_1K_1L_1$ and $A_1B_1C_0D_1E_0F_1G_1H_1FxI_0I_1J_1K_1L_0$), respectively. In SNR method, we obtain the best SNR value as 11.9070 in RF_X solution $B_1C_0D_0E_1F_1G_1H_0FxI_0I_1J_0K_0L_1$. So, we can say that we achieved some better solutions as well as some worse ones compared with Karabulut's solutions. Common factors of Karabulut's and our results are generally at the same level. An appropriate way to compare the performance of the two approaches is to collect the data at the optimal solutions of these two approaches separately and then analyze which of the two data obtain closer to the mean and variance of the classes.

CHAPTER 4

DISCUSSION

In the light of this study, we focus on obtaining optimal parameter settings of a product/process the responses of which are ordered as categorical values. Our proposed method has been tested on three case problems, and for all cases, the best solutions, i.e. parameter designs, have been found. By means of the weighting property of the Random Forest method, it has been possible to achieve the desired target values or values very close to those targets for each case. It has also been possible to obtain desirable results in terms of the response variance. Since each case is different than each other, the method shows different performance for each case.

Optimal solutions for all cases are given in Table 4.1. The table provides results of performance measures EV, Var and SNR at the optimal solutions obtained according to optimization of each of them separately. In other words, each EV value in the table is the EV at the optimal solution of the ε-Constraint method that gives the best EV; similarly each Var value in the table is the variance values at the optimal solution of the ε-constraint method that gives the minimum Var value; and each SNR value in the table is the SNR values at the optimal solution of maximization of SNR problem for related cases. Making general inferences from this table is not easy but if we are to make an evaluation for only our cases:

• The optimal solutions according to the expected value generally emerge from the models whose weights are exponentially distributed,
• The optimal solutions according to the variance are generally observed in the models whose weights are exponentially distributed,
• The optimal solutions according to SNR values generally appear from the models whose weights are exponentially distributed.

As mentioned before, we can give weights according to our preference, but these weights would always change depending on problems or the decision maker. Furthermore, it is not the right approach to make generalizations based only on these results. These results are valid only for the cases we used. To be on the safe side, for our cases, to choose exponentially weighted Random Forests might be logical since the optimal results emerge from generally in that.

When we want to analyze the relation between weighting strategies and differences between the expected value and target, we can give a graph (Figure 4.1) of the differences between the expected value and target value for each case and each weighting strategy.

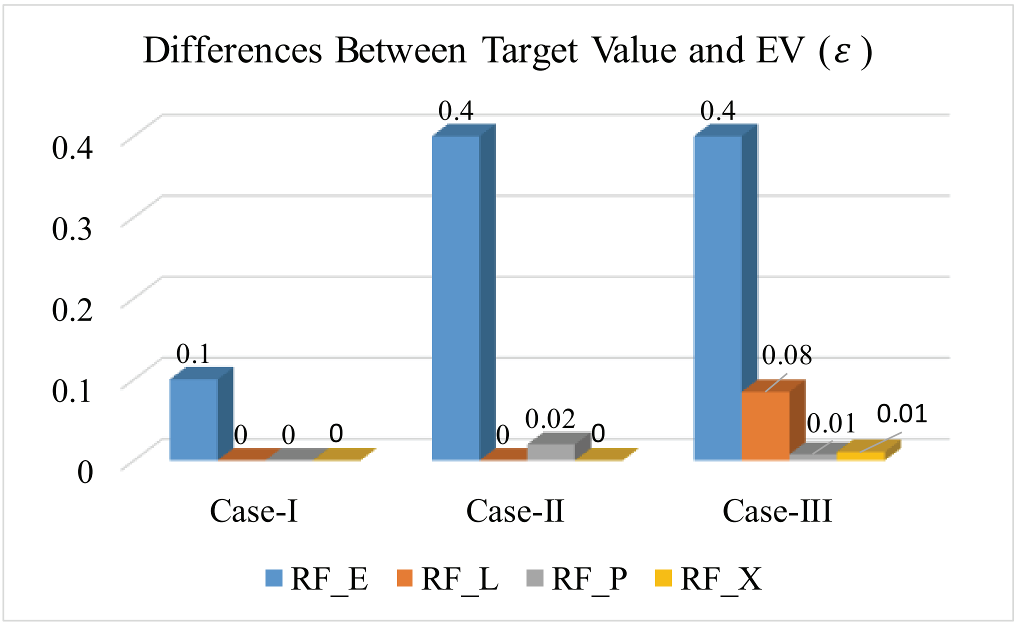


Figure 4.1. Differences Between Target Value and the Expected Value for Each Case

According to the Figure 4.1, it is clearly seen that giving weights to the classes, especially giving the highest weight to our class of interest, helps us converge to our target in a more precise fashion.

Table 4.1. Overall Comparison Table According to the Best Results for All Cases

Problem Type/ Desired Class	Cases	Equal Weight			Linear Weight			Piecewise Weight			Exponential Weights		
		$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$	$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$	$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$	$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
STB[1]/ I	Surface Defect	🟦 1	2×10^{-6}	🟩 0	🟦 1	1×10^{-5}	🟩 0	🟦 1	1×10^{-5}	-0.01	🟦 1	🟧 1×10^{-7}	🟩 0
LTB[2]/ IV	Inkjet Printer	3.66	1.060	10.65	🟦 4	5.4×10^{-8}	12.03	3.98	🟧 -1.4×10^{-8}	12.03	🟦 4	0.06	🟩 12.04
LTB/ IV	Dupli-cator	3.61	0.458	10.77	3.92	0.07	11.77	🟦 3.99	0.03	11.84	3.99	🟧 0.01	🟩 11.91

[1] STB: Smaller-the-Better

[2] LTB: Larger-the-Better

🟦 Optimal expected values for the cases

🟧 Optimal variance values for the cases

🟩 Optimal SNR values for the cases

The graph of variance is given in Figure 4.2. According to this graph, while we give a higher weight to our class of interest, the variance value of the model decreases (Here, in some cases, variances are sometimes negative, but since they are very close to zero, we can assume these values to be equal to zero).

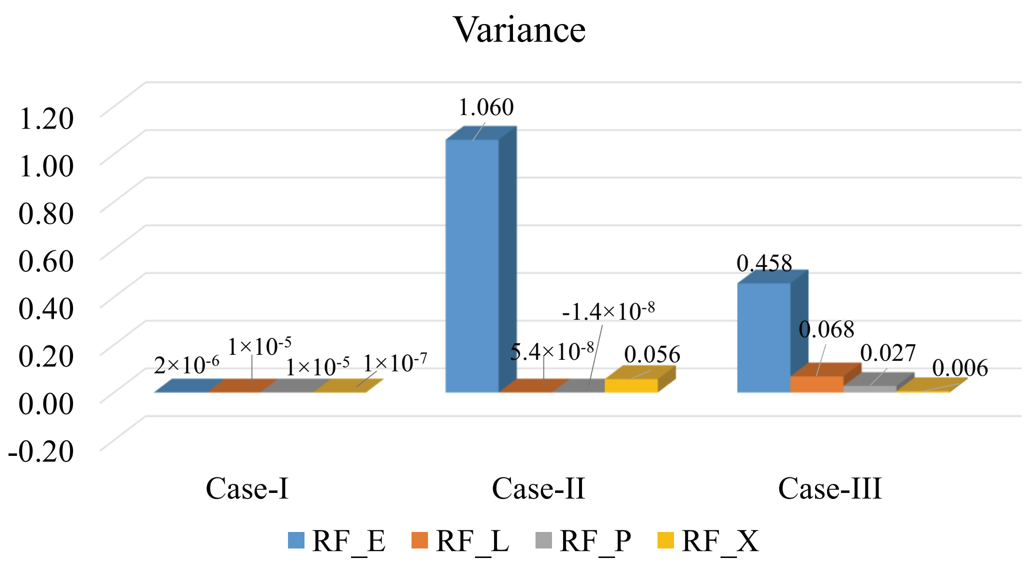


Figure 4.2. The Best Variance Value for Each Case

Furthermore, when the results of the cases are examined, it is seen that all Random Forest models (RF_E, RF_L, RF_P, and RF_X) in the Surface Defect case achieves the optimal expected value, that is 1. Also, optimal variance values are generally very close to optimal value, that is 0, for all Random Forest models in the Surface Defect case compared with the other cases. The Inkjet Printer Case exhibits a slightly better performance than the Duplicator Case, since, in the Inkjet Printer case, the estimated expected values are obtained at its optimal value, that is 4, for some weighted models, while in Duplicator Case, the estimated expected value has approached to 4 but not exactly attained this value. To find the reason of this situation, we analyze some features of data sets; sizes, the number of factors and classes, etc., and the results of the analysis is given in Table 4.2. When we examine the table, it is seen that the data size of the Surface Defect Case is larger than other cases. Also, in this case, the data has a relatively more balanced distribution with regard to classes. In addition to that, in this case, the largest number of observations are under our class of interest. On the other hand, the Inkjet Printer case has the second largest number of observations, while this number is minimum for the Duplicator Case. Moreover, the distribution of the observations on classes is slightly more balanced in the Inkjet Printer case than in the Duplicator Case.

From these views, we can say that the size of data and distribution of the classes have a considerable importance for the Random Forest method. In other words, the Random Forest method runs efficiently, when the size of data is large and data is balanced.

One of the questions that arises while using Random Forest is how to decide the weights. Indeed, weights are determined according to the opinion of the decision makers. Based on each problem and who the decision maker is, the importance of the classes and, in parallel, weights of the classes are changed. However, in our study, there are no decision makers, but we choose three (total four different strategies when we add the Random Forest with equal weight, i.e. RF_E) different methods that are based on different distributions (linear, piecewise linear and exponential). However, if you need advice as to how to determine the weights of classes, you can follow one of the methods below:

1. The decision maker can specify the importance of the classes by using AHP (Analytic Hierarchy Process), and then he/she can use the results of it.
2. The decision maker can decide the weights of the classes according to sensitivity levels of the weights. So, if the result changes significantly, even if the weights of classes exhibit small changes, the decision maker can decide on weights according to these critical points.
3. If weights are very sensitive, by doing confirmation tests, the decision maker can decide on weights by conducting confirmation tests, if possible.

Table 4.2. Features of data sets

	Data Size			Number of Factors	Number of Classes	Type of problem	Target Class	Percentage of observations for each class				
	Number of different designs	Number of replications	Total number of experiments					I	II	III	IV	V
Surface Defect	18	9	162	6	5	STB	I	30%	17%	17%	14%	22%
Inkjet Printer	8	10	80	5	4	LTB	IV	48%	17%	14%	21%	-
Duplicator	16	4	64	13	4	LTB	IV	17%	55%	6%	22%	-

In addition, non-linear multi-objective optimization is one of the methods we used while finding optimal parameter design of products (or processes). For this method, we apply the ε-Constraint Method. By using this approach, we could solve our problems by converting our two-objective problem (desired expected value and minimum variance) to a one-objective problem. We try to minimize the variance at the objective function, and in the constraint, we try to provide a difference between the desired target value and the expected value as small as possible (in our study, to get more robust solutions, we added the constraint related to the expected value to the objective function with a very small coefficient). However, there emerges yet one more parameter to decide on, and this parameter is ε. This situation changes from case to case. In cases where we have to get the adequate fits, ε becomes 0; in our study, ε reaches about 0.7 for some cases. In that stage, to decide on the value of ε, firstly, we try to obtain the smallest value of ε, where the model is feasible (here, ε is considered as the decision variable instead of the parameter). Then, we try to choose the smallest ε value and the closest values to it. Here, a new question may arise: why do not we take ε as the decision variable from the beginning of the study, instead of taking as the parameter? The answer to this question is, as we mentioned at the beginning of the study, to obtain different solutions for our different aims (desired expected value and minimum variance) and to compare them, in order to provide different solutions for the decision maker for his/her different targets.

Furthermore, after applying the method, we also need to check the performance of the method. If the method we use were a parametric method, data would be expected to fit a distribution. So, the prediction power of the model could be tested by using prediction intervals, confidence intervals, and confirmation intervals. However, since our method is non-parametric, the data do not have to fit a certain distribution, and for that reason, we cannot use these intervals for evaluation. But, we should test the power of the method. In the study, before examining all cases, we use some performance indicators that are commonly used in almost all data mining techniques. However, for an unseen data, we can also analyze the power of the model in the following ways:

1. **1.** Firstly, the model is controlled; whether there are some mistakes or training is done properly. After that, by changing the weights, it is controlled again and its prediction power is evaluated.
2. **2.** Cross validation measures can be used. We can look at the power of train and test sets separately (this power value could be an indicator of accuracy which gives the correctly classified percentage of classes). And, the value below is expected to be smaller than or equal to a threshold value that is decided on by the decision maker. If the value is worse than this threshold value, we can make changes on weights until reaching the desired threshold value.

$$\frac{\left|P_{\text{Train}} - P_{\text{Test}}\right|}{P_{\text{Train}}} * 100 \leq \text{THRESHOLD}$$

3. **3.** We can insert this unseen data to the train set, and we train again; after that, we can change the weights until achieving the desired threshold value.
4. **4.** For the cases where $\left|P_{\text{Train}} - P_{\text{Test}}\right|$ value is very large, Steinberg (2017) suggested the following:
   - We can change the threshold value
   - We can grow a smaller forest
   - We can remove some predictors that affect the model very strongly

Moreover, we can develop the model, and if it is possible, we can make the data bigger. These manipulations can decrease the differences between the train and test set and, accordingly, it can strengthen the model.

Finally, the method we use; that is Random Forest, is, as a matter of fact, a method that is easy to implement in many ways. It is applied to all types of data, because of being a non-parametric method. This is one of the features that facilitates our work. Being applicable to data both the responses and the parameters of which are continuous or categorical is another important feature of our method. Removing the imbalance by giving weights allows us to test the cases for different situations easily. Moreover, to be able to make analyses, many software packages enable us to compare the cases in a different environment.

Within the scope of this study, R-Studio, which gives us probability values, weights, images, graphs, etc., is preferred to provide integrity for all analyses. Also, during the study, Random Forest is tested with Rapid Miner, Salford System, WEKA, and MATLAB for testing. However, there have been difficulties encountered in implementing the method. For example, the number of trees that should be in the forest does not have a certain rule, so the best number is obtained as a result of many experimentations. It sometimes takes too much time, and there is no mechanism to test whether it reaches the best value. Secondly, as pointed out, weights are a decision that should be made by the decision maker. Since they can be quite varied, many different strategies can be developed. These changes may affect the results. Lastly, since Random Forest contains hundreds or even thousands of independent trees, it does not give all single trees as an output. This causes us not to be able to follow the developed algorithm behind the method completely.

CHAPTER 5

CONCLUSION AND FURTHER STUDIES

In this study, we propose a new Robust Parameter Design method for products and processes with a single ordered categorical response. For that, we try to find the best parameter settings based on the ε-Constraint and the SNR methods by using the probabilities, expected values, and variances obtained from Random Forests. The method is investigated by applying it on three cases. After that, we have compared our results with those of Karabulut (2013) obtained by using Logistic Regression and other four different ANOVA-based methods. Moreover, now that our proposed method is based on a data mining technique, we analyze the performance of our method with the common criteria which is used to observe the performance of almost all data mining techniques. According to our aim, we are able to give the weights to classes as a result of the advantage that is provided to us by the Random Forest method. Three different weighting strategies other than equal weighting strategies are applied to all cases, and then their results are compared with those of the unweighted Random Forest.

In every case used in our study, we observe that when we use the exponential distribution strategy for our class of interest to which we assign the largest weight among other classes, we can estimate our class of interest with the highest probability, and obtain an expected value that is very close (or equal) to the value we desire. With this weighting method, we are able to reduce the misclassification rate of our class of interest as much as possible.

The proposed method is tested on three different cases. One of them is Smaller-the-Better type problem, and the other two are Larger-the-Better type problems. There is no evidence whether the method performs better for any type of problem. The number of classes and the distribution of these are also different for each case. In some cases, classes have a closer number of observations (slightly more balanced data set), while in some cases there are considerable differences in the number of observations among the classes. We apply the same weighting strategies to all cases, and then, observe the differences of the strategies and how these differences affect the performance of the method. We realize that the Random Forest shows better performance in slightly more balanced data sets. In addition, in the cases we used, it is seen that the number of trees used in the forest does not affect the results significantly. However, for the sake of further research, the number of trees that should be used in the forest can be studied. Moreover, for the data sets which have the missing value, the internal properties of the Random Forest algorithm can be used. It must be underlined that we chose not to exhaust the hypothesis in our study. Moreover, data sizes, the number of parameters, the number of classes and the number of experimental runs used in data are different from each other as well. But there are no signs regarding the effect of these factors on the performance of the method to make an overall assessment. Also, since Random Forest is a random algorithm, it can be possible to obtain different results for different runs. In our study, Random Forest has been rerun numerous times to obtain better (low) OOB error. Furthermore, we could obtain different alternative solutions by changing the value of ε in the ε-constraint method. According to the types of problems, no performance difference has been observed in either optimization method (ε-Constraint and SNR methods). Therefore, the proposed method can be analyzed with more problems, and the performance of all these properties and methods against the problem types can be examined. Both of our optimization methods are important, because they evaluate both the location (mean) and the dispersion effect (variance) separately. In addition to that, these optimization methods are applicable to many common software programs.

Moreover, our proposed method is a strong approach in many ways. There is no importance of parameters and responses being categorical or continuous, and the method is applicable to all sorts of the data types. Since our method is a tree-based method, it is a data hungry method as is the case for Decision Tree. Therefore, a more accurate prediction requires more trees which might result in the slower model and require more memory. However, it can also show satisfactory performance for small data (the method split the data as the train and test by itself. This is the usable property for the small data sets). But, of course, it performs better in large data sets. Conducted experiments prove that the accuracy of the Random Forest method reaches 97-98% in large data sets. In most real-world applications the Random Forest algorithm is fast and simple enough, but there can certainly be situations for which it is hard to use Random Forest. For example, there is no rule to decide some parameters that are used during applying Random Forest. These parameters are number of split value, number of trees used in the algorithm, and weights that are given to classes. The values of these parameters are change based on data type and decision maker. For the number of split value, Breiman (2001) offers some rules, but he also emphasizes that these rule can be changeable according to data set used. For the number of trees used in the forest and weights that are given to classes, there is no rule to decide. These parameters are based on the data set and also need to be decided by the user. Because of these properties, one can prefer to not use the Random Forest method.

Our study, which offers a new method for Robust Parameter Design problems, is not a comparative study. Instead of showing that Random Forest algorithm can give better results than the other algorithms, the main objective of the study is to show that Random Forest use can also provide meaningful results. The sample cases for which LR shows significant results are selected. Moreover, by selecting the cases where LR shows a relatively weak performance, it can be shown whether the Random Forest algorithm performs better or not. Furthermore, the performances of the methods can be compared using the examples where the Random Forest can be weak and LR can be more successful. For a meaningful comparison, the variables can be accepted on the original scales in both approaches because both methods are suitable for both types of variables. Since we do not aim to make comparisons, we have set the variables in their original values using the properties of Random Forest.

Furthermore, we apply our proposed method to only Smaller-the-Better and Larger-the-Better types of problems, which have the ordinal responses, but there also exists

Nominal-the-Best types of problems. We did not have the opportunity to investigate such problems in our study, but Erdural (2006) gives some advice for such types of problems. First, we can set our class of interest as the preferred class, and the others can be set as non-preferred ones. That way, we can convert a nominal case to a binary case. The other suggestion is to convert the nominal case to an ordinal case by ranking the responses upon our wish. For both cases, our method would work; so, in further studies, our method can be applied to Nominal-the-Best cases.

Finally, Random Forest is an ensemble method which is unique to the decision tree. The software we use (R-Studio) generates the trees in the forest according to CART algorithm. In further studies, other decision tree methods (ID3, C4.5, C5.0) can be used, and whether or not the type of tree affects the performance of the method can be investigated. For Robust Design studies, different data mining techniques can be applied to the ensemble. The ensemble of Random Forest and different data mining techniques can even be a double ensemble.

As a result, Robust Design studies about categorical responses are few, albeit they are increasing in number day by day. Since the Random Forest method has not been used before in such problems, our study contributes to the area of Robust Parameter Design.

REFERENCES

Aggrawal, C. C. (2015). *Data Classification: Algorithms and Applications.* Taylor&Francis Group, Florida, USA.

Ardakani, M. K., Noorossana, R., Niaki, S., T., A. and Lahijanian, H. (2009). "Robust Parameter Design Using The Weighted Metric Method—The Case Of 'The Smaller The Better" *Int. J. Appl. Math. Comput. Sci. 19(1),* 59–68.

Asiabar, M.H., & Ghomi, S. M. T. F. (2006). "Analysis of Ordered Categorical Data Using Expected Loss Minimization". *Quality Engineering. 18(2),* 117-121.

Biau, G. (2012). "Analysis of Random Forests Model" *Journal of Machine Learning Research 13*, 1063-1095.

Box, G., & Jones, S. (1986). Discussion of "Testing in industrial experiments with ordered categorical data" *Technometrics, 28(4)*, 295-301.

Breiman, L. (1996). "Bagging Predictors" *Machine Learning, 24*, 123–140.

Breiman, L. (1999). "Random Forests-Random Features" Technical Report 567.

Breiman, L. (2001). "Random Forests" *Machine Learning, 45*, 5-32.

Breiman, L. (2003). "Manual for Setting Up, Using, and Understanding Random Forest"

Breiman L., Cutler, A. & Freidman (2012). "Introduction to Random Forests", *SPM Users Guide.*

Breiman L, Cutler A. (2017). “Random forests – Classification manual.”
[cited; Available from:
https://www.stat.berkeley.edu/~breiman/RandomForests/
Last accessed date: 21 January 2018.

Brodersen, K. H., Ong, C.S., Stephan, K.E. and Buhmann, J.M (2010). “The Balanced Accuracy and Its Posterior Distribution”, *10 Proceedings of the 2010 20th International Conference on Pattern Recognition (ICPR),* 3121-3124.

Chipman, H., & Hamada, M. (1996). “Bayesian analysis of ordered categorical data from industrial experiments.” *Technometrics. 35(1),* 1-10.

Cohen J. A (1960). “Coefficient of agreement for nominal scales”, *Educ. Psychol Meas, 20(1),* 37–46.

Eldardiry, H. & Neville, J. (2011). “Across-Model Collective Ensemble Classification”, *Twenty-Fifth AAAI Conference on Artificial Intelligence,* 343-349.

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.

Fawcett, T. (2006). “An Introduction to ROC Analysis”, *Pattern Recognition Letters 27,* 861-874.

Ferri, C., Hernández-Orallo, J., Modroiu, R. (2009). “An Experimental Comparsion of Performance Measures for Classification” *Pattern Recognition Letters 30,* 27-38.

Friedman, J., Hastie, T., Rosset, S., Tibshirani, R. and Zhu, J. (2004). “Discussion of three boosting papers by Jiang, Lugosi, and Vayatis, and Zhang.” *Annals of Statistics 32*, 102–107.

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.

Gravetter, F. J., & Wallnau, L. B. (2014). *Essentials of statistics for the behavioral sciences.* Belmont, CA : Wadsworth Cengage Learning.

Gupta, C., Toda, H., Mayr, P., and Sommitsch, C. (2015). "3D creep cavitation characteristics and residual life assessment in high temperature steels: a critical review." *Materials Science and Technology*, *31*(5), 603-626.

Haimes, Y. Y., Lasdon, L. S., and Wismer, D. A. (1971). "On a bicriterion formulation of the problems of integrated system identification and system optimization." *IEEE Transactions on Systems, Man, and Cybernetics, 1,* 296– 297.

Han, J., Kamber, M. and Pei, J. (2012). *Data Mining: Concepts and Techniques.* Elsevier, 225Wyman Street, Waltham, MA 02451, USA.

Hastie, T., Tibshirani, R. & Friedman, J. (2008). *The Elements of Statistical Learning; Data mining, Inference and Prediction.* Springer Verlag, New York.

He, Yan (2006). *Missing Data Imputation for Tree-Based Models*/ Unpublished Ph.D. Thesis, University of California, Los Angeles, USA.

Jeng, Y. C., & Guo, S. M. (1996). "Quality improvement for RC06 chip resistor." *Quality and Reliability Engineering International, 12(6),* 439-445.

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.

Köksal, G., Erdural, S., and 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.

Kuncheva, L. (2014). *Combining Pattern Classifier.* John Willey & Sons, Inc. Hoboken, New Jersey, USA.

Lee, J. W., Lee, J.B., Park, M., and Song, S., H. (2005). “An extensive comparison of recent classification tools applied to microarray data.” *Computational Statistics & Data Analysis, 48,* 869-885.

Li, C. (2013). "Probability Estimation in Random Forests" *All Graduate Plan B and other Reports,* Paper No:312

Lin, N., Wu, B., Jansen, R., Gerstein, M., and Zhao, H. (2004). “Information assessment on predicting protein-protein interactions.” *BMC Bioinformatics*, 5, 154.

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

Logothetis, N. & Wynn, H., P. (1994). *Quality Through Design.* Oxford University Press Inc., New York, USA.

Machado, G., Mendoza, M.R. and Corbellini, L.G. (2015). “What variables are important in predicting bovine viral diarrhea virus? A random forest approach.” *Veterinary Research*, *46 (1),* 85.

MathWorks, MATLAB (2013).
https://www.mathworks.com/

Mercy, I. (2012). Data Warehousing and Mining Engineering Lecture Notes- Ensemble Learning and Model Selection.
[cited; Available from:
http://shareengineer.blogspot.com.tr/2012/09/ensemble-learning-and-model-selection.html
Last accessed date: 28 May 2018.

Miettinen, K. (2012). *Nonlinear Multiobjective Optimization* (Vol. 12). Springer Science & Business Media.

Minitab 18 Statistical Software. (2018). [Computer Software]. State Collage, PA: Minitab Inc. Retrieved from www. minitab.com

Montgomery, D. C. (2009). *Introduction to Statistical Quality Control*. Wiley, New York, USA.

Montgomery, D. C. (2013). *Design and Analysis of Experiments*. Wiley, New York, USA.

Muchlinski, D., Siroky, D., He, J. and Kocher, M. (2015). “Comparing Random Forest with Logistic Regression for Predicting Class-Imbalanced Civil War Onset Data.” *Political Analysis 24,* 87–103.

Myers, R. H., Montgomery, D. C. and Anderson-Cook, C. M. (2009). *Response Surface Methodology: process and product optimization using designed experiments*. Wiley, New Jersey, USA.

Nair, V. N. (1986). “Testing in industrial experiments with ordered categorical data” *Technometrics, 28(4),* 283-291.

Okun, O., & Valentini, G. (2009). *Applications of Supervised and Unsupervised Ensemble Methods.* Springer-Verlag Berlin Heidelberg. A Wiley-Interscience Publications, New York, USA.

Phadke, M.S. (1989). *Quality engineering using robust design.* New York: Prentice-Hall, USA.

Phadke, M. S., & Dehnad, K. (1988). "Optimization of Product and Process Design for Quality and Cost" *Quality and Reliability Engineering International*, *4*(2), 105-112.

Pham, H. (Ed.). (2006). *Springer handbook of engineering statistics*. Springer Science & Business Media.

Quach, A. T. (2012). "Interactive random forests plots" *All Graduate Plan B and other Reports,* Paper 134.

R Core Team. (2016). R: A language and environment for statistical computing. R Foundation for Statistical Computing. Vienna, Austria. Retrieved from http://www.r-project.org/

Read, T. R. C., & Cressie, N. A. C. (1988). *Goodness-of-fit statistics for discrete multivariate data.* New York: Springer-Verlag, USA.

Rokach, L. (2010). *Pattern Classification Using Ensemble Methods.* World Scientific Publishing Co. Pte. Ltd., 5 Toh Tuck Link, Singapore.

Ruiz-Gazen, A. & Villa, N. (2007). "Storms Prediction: Logistic Regression Vs Random Forest For Unbalanced Data" *Cases Studies in Business, Industry and Government Statistics (CSBIGS), 1 (2),* 91-101.

Steinberg, D. (2017). "Differences Between Train and Test Performance Results". Salford Systems.
[cited; Available from:
https://www.salford-systems.com/blog/dan-steinberg/differences-between-train-and-test-performance-results
Last accessed date: 21 January 2018.

Sun, Y., Kamel, M. S., Wong, A. K. C. and Wang, Y. (2007). "Cost-sensitive boosting for classification of imbalanced data." *Pattern Recognition 40*, 3358-3378.

Wendell, F. Smith (2005). *Experimental Design for Formulation.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, Pennsylvania, USA.

Witten, I.H., & Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques.* 500 Sansome Street, Suite 400, San Francisco, CA 94111, USA.

Wu, F., & Yeh, C. (2006). "A comparative study on optimization methods for experiments with ordered categorical data". *Computers & Industrial Engineering. 50(2006),* 220-232.

Yoo, W., Ference, B.A., Ference, B.A., Cote, M.L. and Schwartz, A. (2012). "A Comparison of Logistic Regression, Logic Regression, Classification Tree, and Random Forests to Identify Effective Gene-Gene and Gene-Environmental Interactions." *National Institutes of Health, 2012 August; 2(7)*: 268-294.

Zhang, C., & Ma, Y. (2012). *Ensemble Machine Learning: Methods and Applications.* Springer New York Dordrecht Heidelberg, London, UK.

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

APPENDIX A

PSEUDOCODE OF ALGORITHMS

```
$T$ is number of iteration
$S$ is the original data set
$L$ is the sample size
$M$ is the class that is the result of a used classification method

Step 1: $t \leftarrow 1$
Step 2: $S_t$ a sample of L instances from S with replacement
Step 3: Construct classifier $M_t$ with $S_t$ as the training set
Step 4: $t \leftarrow t+1$
Step 5: until $t \leftarrow T$

Let,
$x$ is an instance to be classified
C is predicted class

1. Initially, set votes for all class as 0
2. For $i$=1 to $T$
3. $vote_i \leftarrow M_i(x)$ {get predicted class from member $i$}
4. Increase by 1 the counter of corresponding class
5. $C \leftarrow$ the class with the largest number votes
```

Figure A.1. Pseudocode of Bagging Algorithm

```
1. For $b$ = 1to $B$ ; number of trees:

   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size $N$ from the training data
   (b) Grow a random-forest tree $T_b$ to the bootstrapped data, by recursively
       repeating the following steps for each terminal node of tree, until the
       minimum node size $n_{min}$ is reached.

       i.    Select $m$ variables at random from the $p$ variables.
       ii.   Pick the best variable / split-point among the $m$.
       iii.  Split the node into two daughter nodes.

where $p$ is the total number of variables (parameters) of the product/process
2. Output the ensemble of trees $\{T_b\}$.
   To make a prediction at a new point $x$:

   Regression: $\hat{f}_{rf}^{B}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$.

   Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random
   forest tree when $x$ is passed down the bth random forest tree.
Then $\hat{C}_{rf}^{B}(x) =$ majority vote $\left\{\hat{C}_b(x)\right\}^{B}$
```

Figure A.2. Pseudocode of Algorithm of Random Forest for Regression and Classification

APPENDIX B

PERFORMANCE MEASURES OF CLASSIFICATION METHODS

Table B.1. Performance Measures of Classification Methods and Their Definitions and Formulas

Measure	Definition	Formula	References
Accu-racy	Right classification rate	$\text{Acc} = \dfrac{\text{Total Number of Correctly Classified Observations}}{\text{Size of the Data Set}}$	
Mis-class. Error	Wrong classification rate	MisErr=1-Acc	
AUC	Area under the ROC curve	$AUC = \int_a^b d\left(ROC\right)$	Fawcett (2006)

Table B.1 (cont'd). Performance Measures of Classification Methods and Their Definitions and Formulas

Measure	Definition	Formula	References
F-Score	Harmonic mean of your precision and recall	$\mathrm{F-Score} = \frac{\sum_{k=1}^{K} \mathrm{F-measure}(k)}{\mathrm{K}}$ $\mathrm{F-measure(k)} = \frac{2 * \mathrm{recall}\ (k) * \mathrm{precision}\ (k)}{\mathrm{recall} + \mathrm{precision}}$	Han et al.(2012)
G-Mean	Geometric mean of your precision and recall	$\mathrm{G-Mean} = \sqrt[\mathrm{K}]{\prod_{k=1}^{K} \mathrm{recall}(k) * \mathrm{precision}\ (k)}$	
Kappa	Agreement between two classifiers	$\mathrm{Kappa} = \frac{\mathrm{Acc} - \theta}{1 - \theta}$; N: Size of the Data Set $\theta = \frac{\sum_{k=1}^{K} \text{Number of Observation in Class } k * \text{Number of Observation Classfied as } k}{\mathrm{N}^2}$	Cohen (1960) Witten and Frank (2005)

Table B.1 (cont'd). Performance Measures of Classification Methods and Their Definitions and Formulas

Measure	**Definition**	**Formula**	**References**
Precision	The fraction of retrieved instances that are relevant for the considered class (class based measure)	$\text{Precision}\ (k) = \frac{\text{Total Number of Observation Correctly Classified in Class } k}{\text{Total Number of Observation Predicted as Class } k}$ **A sign of sharpness in determining the class of interest.	Han et al.(2012)
Recall (Sensitivity)	The fraction of relevant instances that are retrieved for the considered class (class based measure)	$\text{Recall (k)} = \frac{\text{Total Number of Observation Correctly Classified in Class } k}{\text{Total Number of Observation in Class } k}$	Han et al.(2012)

Table B.1 (cont'd). Performance Measures of Classification Methods and Their Definitions and Formulas

Measure	**Definition**	**Formula**	**References**
Specificity	True negative rate of the considered class (class based measure)	$\text{Specificity}\ (k) = \frac{\text{Total Number of Observation} \atop \text{that is not in Class k and is not Classified as Class } k}{\text{Total Number of Observation that is not Classified as Class } k}$	Han et al.(2012)

APPENDIX C

RESULTS FOR THE SURFACE DEFECTS CASE

C.1. Variable Importance Plots of Different Random Forests for the Surface Defects Case

RF_E

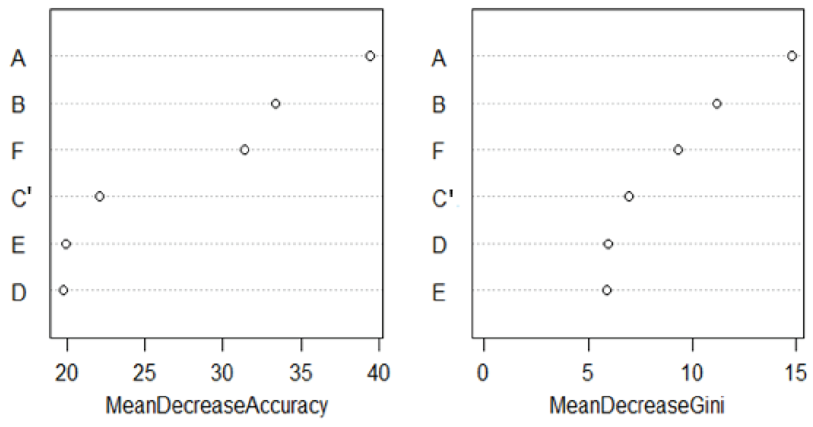


Figure C.1. Variable Importance Plot of RF_E for the Surface Defects Case

RF_L

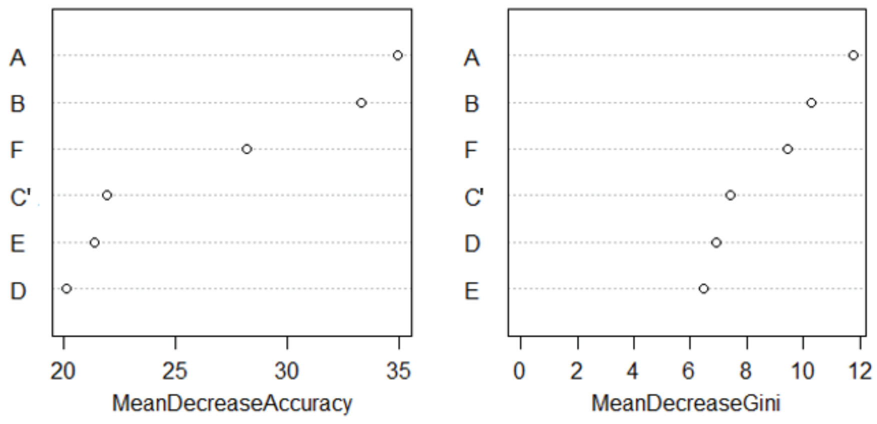


Figure C.2. Variable Importance Plot of RF_L for the Surface Defects Case

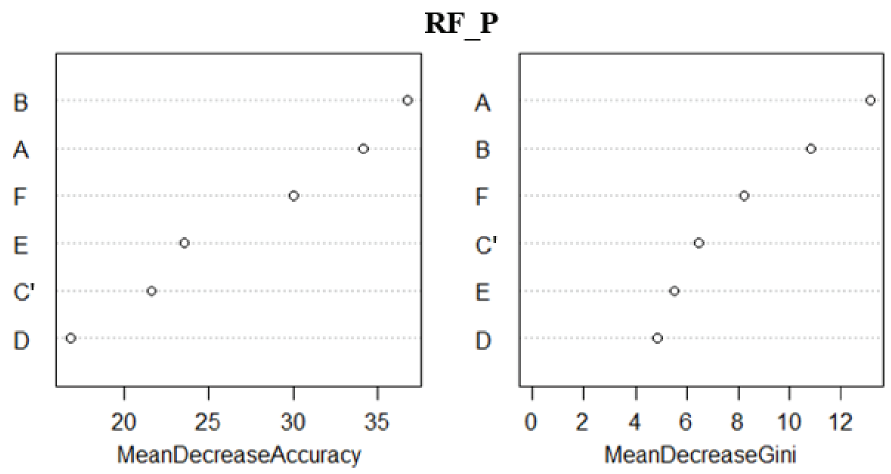


Figure C.3. Variable Importance Plot of RF_P for the Surface Defects Case

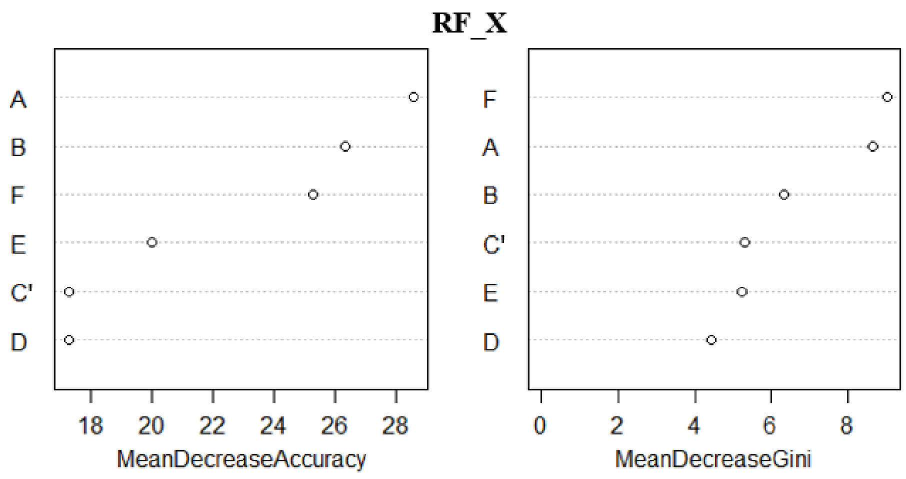


Figure C.4. Variable Importance Plot of RF_X for the Surface Defects Case

C.2. Estimated Probability, Expected Values, Variances and SNR of Random Forests for the Surface Defects Case

Table C.1. Estimated Probability, the Expected Value, the Variance and SNR of RF_E for the Surface Defects Case

Exp. No	Estimated Probabilities from RF_E					$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$	$\hat{P}(Y=V)$			
1	0.9986	0	0	0	0.0014	1.0057	0.0228	-0.1464
2	0.7786	0.1157	0.1057	0	0	1.3271	0.4316	-3.4101
3	0.0257	0	0.9043	0.0700	0	3.0186	0.1725	-9.6775
4	0	0.9971	0.0029	0	0	2.0029	0.0029	-6.0361
5	0.0057	0.0271	0.0014	0.4900	0.4757	4.4029	0.4749	-12.9800
6	0.0314	0.0029	0.6314	0.0343	0.3000	3.5686	1.0396	-11.3910
7	0.0086	0.0471	0.0443	0.5857	0.3143	4.1500	0.6018	-12.5100
8	0.3971	0	0.1571	0.0271	0.4186	3.0700	3.2851	-11.042
9	0	0	0.0029	0.3257	0.6714	4.6686	0.2273	-13.429
10	0.9971	0	0.0029	0	0	1.0057	0.0114	-0.0982
11	0.9914	0.0071	0.0014	0	0	1.0100	0.0128	-0.1404

Table C.1 (cont'd). Estimated Probability, the Expected Value, the Variance and SNR of RF_E for the Surface Defects Case

Exp. No	Estimated Probabilities from RF_E					$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y = I)$	$\hat{P}(Y = II)$	$\hat{P}(Y = III)$	$\hat{P}(Y = IV)$	$\hat{P}(Y = V)$			
12	0.1771	0.3486	0.4400	0	0.0343	2.3657	0.7920	-8.0540
13	0.6114	0.1843	0.1629	0.0343	0.0071	1.6414	0.8471	-5.4918
14	0.1114	0.3100	0.5743	0	0.0043	2.4757	0.4980	-8.2133
15	0.0014	0.0286	0.0129	0.0086	0.9486	4.8743	0.3242	-13.817
16	0.3057	0.5329	0.1400	0.0200	0.0014	1.8786	0.5238	-6.0776
17	0.1886	0.0371	0.0014	0.1714	0.6014	3.9600	2.4470	-12.584
18	0	0.0014	0	0.0443	0.9543	4.9514	0.0548	-13.904

Table C.2. Estimated Probability, the Expected Value, the Variance and SNR of RF_L for the Surface Defects Case

Exp. No	Estimated Probabilities from RF_L					$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y = I)$	$\hat{P}(Y = II)$	$\hat{P}(Y = III)$	$\hat{P}(Y = IV)$	$\hat{P}(Y = V)$			
1	1	0	0	0	0	1	0	0
2	0.6543	0.2171	0.1286	0	0	1.4743	0.5065	-4.2814
3	0.0314	0.0014	0.9214	0.0457	0	2.9814	0.1725	-9.5720
4	0	0.9957	0.0043	0	0	2.0043	0.0043	-6.0438
5	0.0057	0.1771	0.0086	0.7686	0.0400	3.6600	0.6930	-11.4887
6	0.0386	0.0043	0.8929	0.0357	0.0286	3.0114	0.3084	-9.7207
7	0.0043	0.1886	0.0771	0.7114	0.0186	3.5514	0.6874	-11.2385
8	0.5814	0.0029	0.3214	0.0557	0.0386	1.9671	1.4718	-7.2766
9	0.0043	0.0014	0.0029	0.8671	0.1243	4.1057	0.1603	-12.3089
10	0.9986	0	0.0014	0	0	1.0029	0.0057	-0.0494
11	0.9914	0.0086	0	0	0	1.0086	0.0085	-0.1103
12	0.1386	0.5286	0.3314	0.0014	0	2.1957	0.4374	-7.2087
13	0.5986	0.2314	0.1443	0.0257	0	1.5971	0.6834	-5.0978

Table C.2 (cont'd). Estimated Probability, the Expected Value, the Variance and SNR of RF_L for the Surface Defects Case

Exp. No	Estimated Probabilities from RF_L					$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$	$\hat{P}(Y=V)$			
14	0.0643	0.5114	0.4243	0	0	2.3600	0.3590	-7.7295
15	0.0029	0.3443	0.1600	0.1414	0.3514	3.4943	1.6585	-11.4203
16	0.1986	0.7214	0.0800	0	0	1.8814	0.2645	-5.8027
17	0.4271	0.2114	0.0057	0.2757	0.0800	2.3700	2.1188	-8.8850
18	0.0029	0.0057	0	0.4171	0.5743	4.5543	0.3156	-13.2340

Table C.3. Estimated Probability, the Expected Value, the Variance and SNR of RF_P for the Surface Defects Case

Exp. No	Estimated Probabilities from RF_P					$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y = I)$	$\hat{P}(Y = II)$	$\hat{P}(Y = III)$	$\hat{P}(Y = IV)$	$\hat{P}(Y = V)$			
1	1	0	0	0	0	1	0	0
2	0.9671	0.0200	0.0129	0	0	1.0457	0.0693	-0.6553
3	0.1357	0.0057	0.7671	0.0886	0.0029	2.8171	0.6151	-9.3204
4	0.0029	0.9914	0.0057	0	0	2.0029	0.0086	-6.0423
5	0.0243	0.0114	0	0.7071	0.2571	4.1614	0.4954	-12.5073
6	0.2086	0	0.6057	0.0500	0.1357	2.9043	1.4180	-9.9356
7	0.0100	0.0386	0.0171	0.7857	0.1486	4.0243	0.4094	-12.2022
8	0.8071	0	0.0514	0.0400	0.1014	1.6286	1.7935	-6.4794
9	0.0043	0	0	0.6529	0.3429	4.3300	0.2725	-12.7924
10	1	0	0	0	0	1	0	0
11	1	0	0	0	0	1	0	0
12	0.5014	0.2686	0.2214	0	0.0086	1.7457	0.7353	-5.7782
13	0.8857	0.0486	0.0329	0.0329	0	1.2129	0.4304	-2.7908

Table C.3 (cont'd). Estimated Probability, the Expected Value, the Variance and SNR of RF_P for the Surface Defects Case

Exp. No	Estimated Probabilities from RF_P					$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y = I)$	$\hat{P}(Y = II)$	$\hat{P}(Y = III)$	$\hat{P}(Y = IV)$	$\hat{P}(Y = V)$			
14	0.4500	0.2429	0.3057	0.0014	0	1.8586	0.7414	-6.2281
15	0.0029	0.0471	0.0186	0.0586	0.8729	4.7514	0.5411	-13.6393
16	0.6829	0.2729	0.0414	0.0029	0	1.3643	0.3316	-3.4101
17	0.5286	0.0314	0.0014	0.1800	0.2586	2.6086	3.2068	-10.0050
18	0.0029	0	0	0.1386	0.8586	4.8500	0.1618	-13.7446

Table C.4. Estimated Probability, the Expected Value, the Variance and SNR of RF_X for the Surface Defects Case

Exp. No	Estimated Probabilities from RF_X					$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$	$\hat{P}(Y=V)$			
1	1	0	0	0	0	1	0	0
2	0.9557	0.0443	0	0	0	1.0443	0.0423	-0.5418
3	0.3800	0.0143	0.5986	0.0071	0	2.2329	0.9529	-7.7368
4	0.0014	0.9971	0.0014	0	0	2	0.0029	-6.0237
5	0.0271	0.5757	0.0043	0.3800	0.0129	2.7757	1.0654	-9.4300
6	0.5143	0.0029	0.4714	0.0086	0.0029	1.9829	1.0454	-6.9698
7	0.0171	0.5543	0.1114	0.3171	0	2.7286	0.8663	-9.1968
8	0.9443	0	0.0514	0.0043	0	1.1157	0.2309	-1.6900
9	0.0214	0.0143	0.0057	0.9014	0.0571	3.9586	0.3111	-12.0362
10	1	0	0	0	0	1	0	0
11	0.9986	0.0014	0	0	0	1.0014	0.0014	-0.0186
12	0.4929	0.4514	0.0557	0	0	1.5629	0.3575	-4.4716
13	0.9314	0.0614	0.0071	0	0	1.0757	0.0843	-0.9392

Table C.4 (cont'd). Estimated Probability, the Expected Value, the Variance and SNR of RF_X for the Surface Defects Case

Exp. No	Estimated Probabilities from RF_X					$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y = I)$	$\hat{P}(Y = II)$	$\hat{P}(Y = III)$	$\hat{P}(Y = IV)$	$\hat{P}(Y = V)$			
14	0.4857	0.4400	0.0743	0	0	1.5886	0.3907	-4.6453
15	0.0086	0.6043	0.2357	0.0814	0.0700	2.600	0.8400	-8.8081
16	0.5800	0.4143	0.0057	0	0	1.4257	0.2559	-3.5956
17	0.7857	0.1700	0	0.0386	0.0057	1.3086	0.5134	-3.4747
18	0.0129	0.0186	0.0071	0.5771	0.3843	4.3014	0.4906	-12.7859

APPENDIX D

RESULTS FOR THE INKJET PRINTER CASE

D.1. Variable Importance Plots of Different Random Forests for Inkjet Printer Example

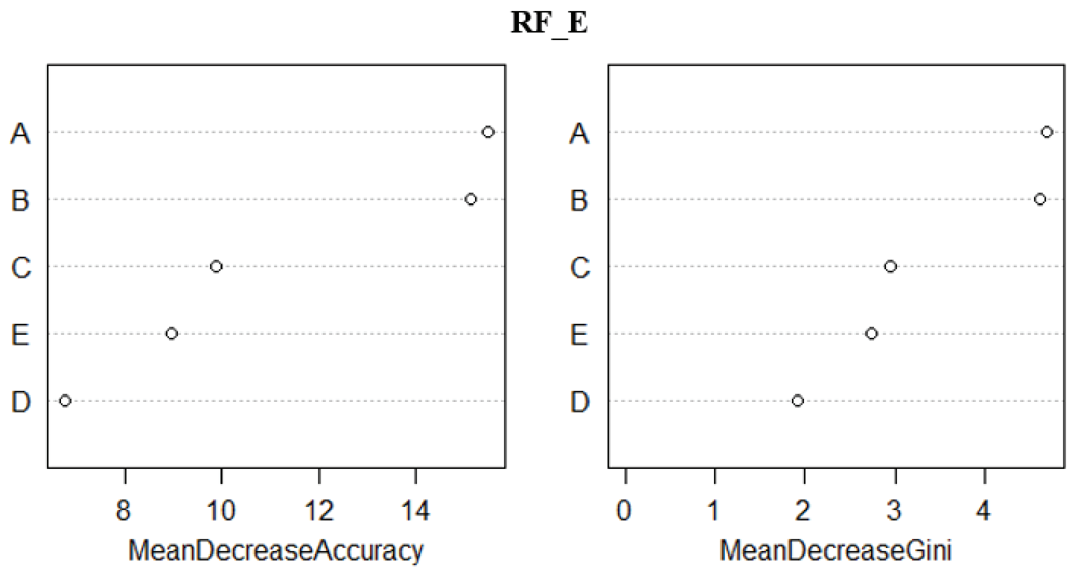


Figure D.1. Variable Importance Plot of RF_E for the Inkjet Printer Case

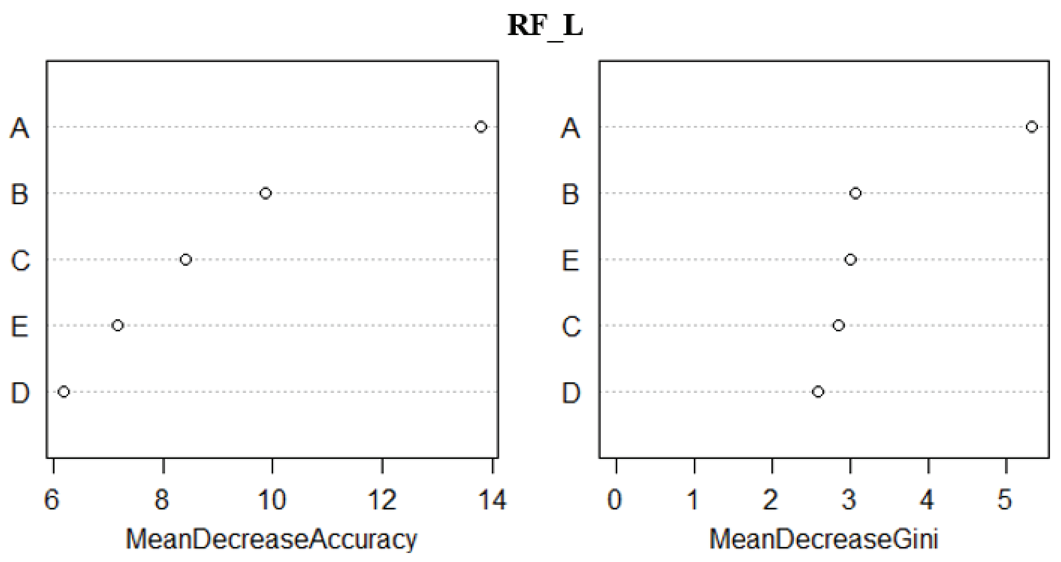


Figure D.2. Variable Importance Plot of RF_L for the Inkjet Printer Case

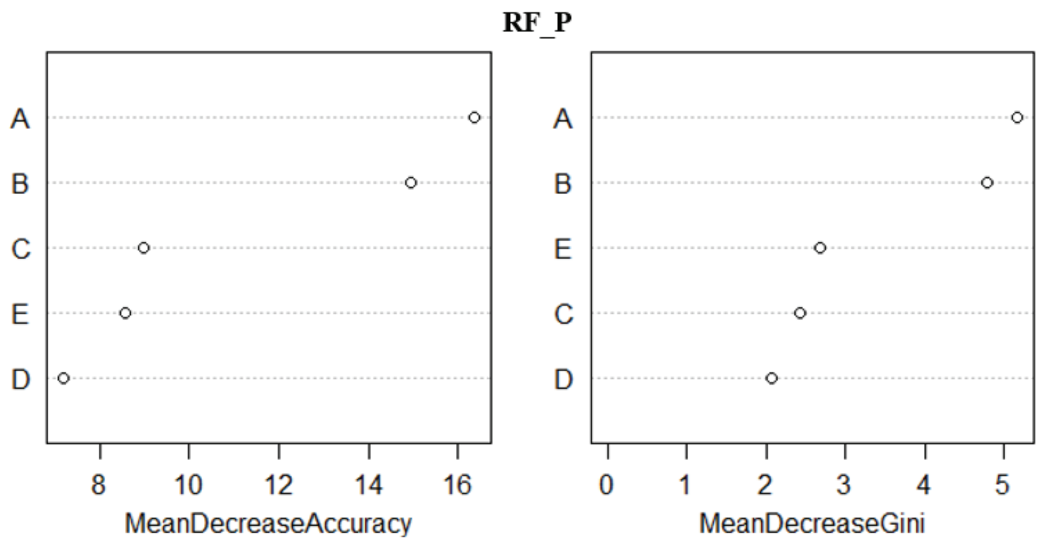


Figure D.3. Variable Importance Plot of RF_P for the Inkjet Printer Case

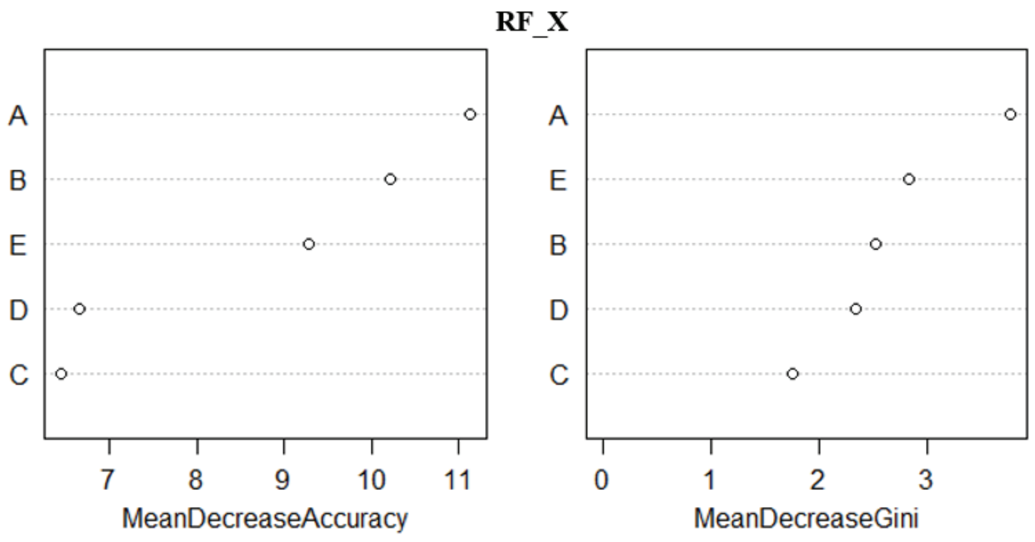


Figure D.4. Variable Importance Plot of RF_X for the Inkjet Printer Case

D.2. Estimated Probability, Expected Values, Variances and SNR of Constructed Different Random Forests for the Inkjet Printer Case

Table D.1. Estimated Probability, the Expected Value, the Variance and SNR of RF_E for the Inkjet Printer Case

Ex. No	A	B	C	D	E	Estimated Probabilities from RF_E				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
						$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$			
1	0	0	0	0	0	0.555	0.240	0.200	0.005	1.655	0.656	2.025
2	0	0	1	1	1	0.225	0.710	0	0.065	1.905	0.476	4.157
3	0	1	0	1	1	0.935	0.020	0	0.045	1.155	0.401	-1.540
4	0	1	1	0	0	0.250	0.535	0.140	0.075	2.040	0.688	4.443
5	1	0	0	0	1	0.980	0.005	0	0.015	1.050	0.138	-0.957
6	1	0	1	1	0	0.995	0	0	0.005	1.015	0.045	-0.403
7	1	1	0	1	0	0.100	0	0.100	0.800	3.600	0.840	10.354
8	1	1	1	0	1	0.170	0.020	0.020	0.790	3.430	1.305	9.458

Table D.2. Estimated Probability, the Expected Value, the Variance and SNR of RF_L for the Inkjet Printer Case

Ex. No	A	B	C	D	E	Estimated Probabilities from RF_L				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
						$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$			
1	0	0	0	0	0	0.005	0.260	0.730	0.005	2.735	0.215	8.380
2	0	0	1	1	1	0.015	0.615	0.090	0.280	2.635	0.822	7.096
3	0	1	0	1	1	0.325	0.255	0.015	0.405	2.500	1.710	5.356
4	0	1	1	0	0	0	0.600	0.290	0.110	2.510	0.470	7.117
5	1	0	0	0	1	0.260	0.005	0.420	0.315	2.790	1.316	7.131
6	1	0	1	1	0	0.770	0.065	0.040	0.125	1.520	1.080	-0.169
7	1	1	0	1	0	0	0	0.200	0.800	3.800	0.160	11.454
8	1	1	1	0	1	0	0	0.030	0.970	3.970	0.029	11.952

Table D.3. Estimated Probability, the Expected Value, the Variance and SNR of RF_P for the Inkjet Printer Case

Ex. No	A	B	C	D	E	Estimated Probabilities from RF_P				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
						$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$			
1	0	0	0	0	0	0.100	0.395	0.500	0.005	2.410	0.452	6.729
2	0	0	1	1	1	0.020	0.560	0.010	0.410	2.810	1.014	7.559
3	0	1	0	1	1	0.575	0.070	0	0.355	2.135	1.977	2.969
4	0	1	1	0	0	0.030	0.525	0.185	0.260	2.675	0.799	7.291
5	1	0	0	0	1	0.560	0.005	0.075	0.360	2.235	2.020	3.536
6	1	0	1	1	0	0.835	0.045	0	0.120	1.405	0.961	-0.957
7	1	1	0	1	0	0	0	0.030	0.970	3.970	0.029	11.952
8	1	1	1	0	1	0	0	0.010	0.990	3.990	0.010	12.011

Table D.4. Estimated Probability, the Expected Value, the Variance and SNR of RF_X for the Inkjet Printer Case

Ex. No	A	B	C	D	E	Estimated Probabilities from RF_X				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
						$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$			
1	0	0	0	0	0	0	0.110	0.885	0.005	2.895	0.104	9.074
2	0	0	1	1	1	0	0.160	0.105	0.735	3.575	0.564	10.525
3	0	1	0	1	1	0.180	0.170	0.015	0.635	3.105	1.514	8.165
4	0	1	1	0	0	0	0.140	0.530	0.330	3.190	0.434	9.553
5	1	0	0	0	1	0.105	0	0.315	0.580	3.370	0.863	9.661
6	1	0	1	1	0	0.775	0.015	0.025	0.185	1.620	1.396	0.048
7	1	1	0	1	0	0	0	0.020	0.980	3.980	0.020	11.982
8	1	1	1	0	1	0	0	0.010	0.990	3.990	0.010	12.011

APPENDIX E

RESULTS FOR DUPLICATOR EXAMPLE

E.1. Variable Importance Plots of Different Random Forests for the Duplicator Case

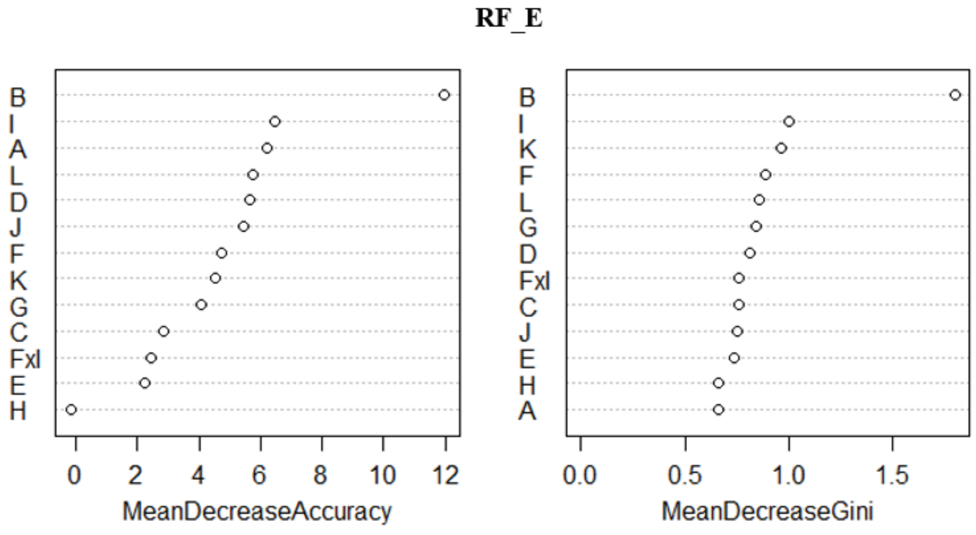


Figure E.1. Variable Importance Plot of RF_E for the Duplicator Case

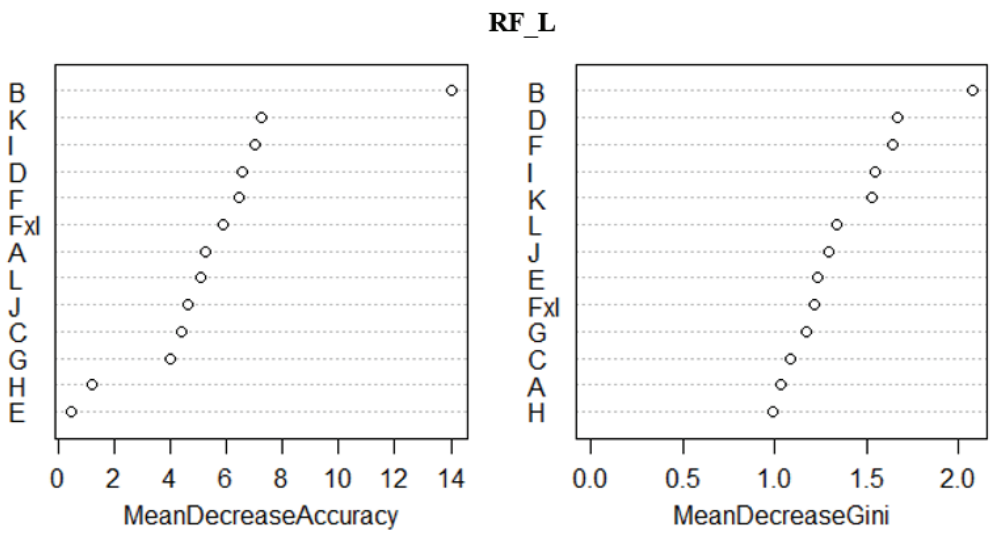


Figure E.2. Variable Importance Plot of RF_L for the Duplicator Case

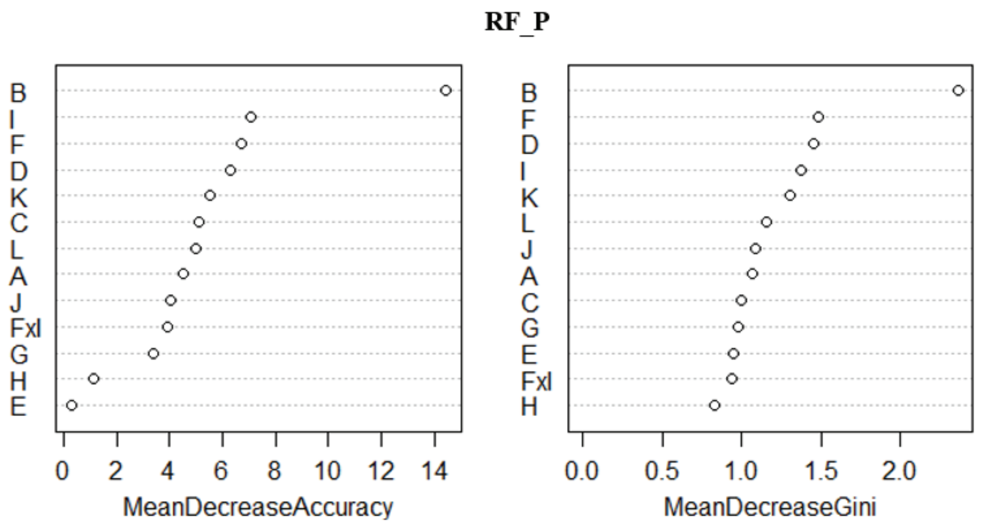


Figure E.3. Variable Importance Plot of RF_P for the Duplicator Case

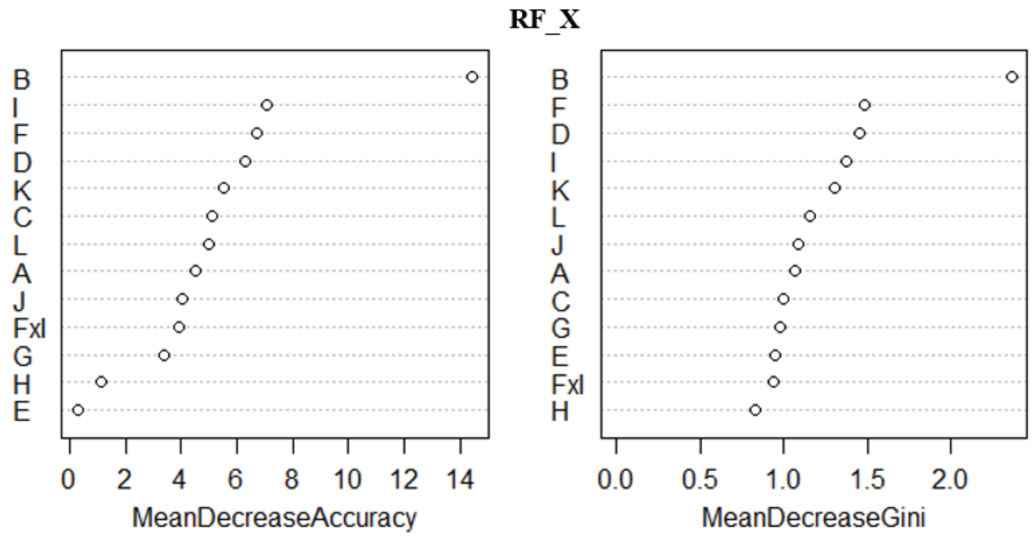


Figure E.4. Variable Importance Plot of RF_X for the Duplicator Case

E.2. Estimated Probability, Expected Values, Variances and SNR of Random Forests for the Duplicator Case

Table E.1. Estimated Probability, the Expected Value, the Variance and SNR of RF_E for the Duplicator Case

Ex. No	Estimated Probabilities from RF_E				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$			
1	0.080	0.865	0.002	0.053	2.028	0.293	5.300
2	0.097	0.837	0.008	0.058	2.027	0.336	5.184
3	0.084	0.851	0.002	0.063	2.044	0.336	5.271
4	0.086	0.846	0.006	0.062	2.044	0.338	5.266
5	0.010	0.816	0.006	0.168	2.332	0.578	6.153
6	0.098	0.295	0.005	0.602	3.111	1.278	8.410
7	0.163	0.463	0.138	0.236	2.447	1.045	5.944
8	0.017	0.703	0.099	0.181	2.444	0.643	6.547
9	0.096	0.853	0.004	0.047	2.002	0.288	5.182
10	0.283	0.636	0.007	0.074	1.872	0.570	3.721
11	0.108	0.836	0.005	0.051	1.999	0.317	5.089
12	0.007	0.952	0	0.041	2.075	0.165	5.867
13	0.018	0.516	0.105	0.361	2.809	0.913	7.678

Table E.1 (cont'd). Estimated Probability, the Expected Value, the Variance and SNR of RF_E for the Duplicator Case

Ex. No	Estimated Probabilities from RF_E				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$			
14	0.019	0.239	0.002	0.740	3.463	0.841	9.960
15	0.015	0.627	0.005	0.353	2.696	0.948	7.181
16	0.105	0.696	0.111	0.088	2.182	0.535	5.516

Table E.2. Estimated Probability, the Expected Value, the Variance and SNR of RF_L for the Duplicator Case

Ex. No	Estimated Probabilities from RF_L				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y = I)$	$\hat{P}(Y = II)$	$\hat{P}(Y = III)$	$\hat{P}(Y = IV)$			
1	0.155	0.506	0.131	0.208	2.392	0.964	5.798
2	0.143	0.508	0.112	0.237	2.443	1.007	5.980
3	0.154	0.475	0.151	0.22	2.437	0.994	5.970
4	0.115	0.498	0.163	0.224	2.496	0.928	6.341
5	0.013	0.311	0.123	0.553	3.216	0.869	9.170
6	0.022	0.019	0.095	0.864	3.801	0.329	11.311
7	0.035	0.077	0.559	0.329	3.182	0.513	9.440
8	0.007	0.084	0.579	0.330	3.232	0.388	9.730
9	0.155	0.486	0.147	0.212	2.416	0.977	5.895
10	0.335	0.297	0.156	0.212	2.245	1.279	4.566
11	0.157	0.492	0.111	0.240	2.434	1.040	5.890
12	0.015	0.683	0.114	0.188	2.475	0.655	6.663
13	0.006	0.026	0.493	0.475	3.437	0.334	10.370

Table E.2 (cont'd). Estimated Probability, the Expected Value, the Variance and SNR of RF_L for the Duplicator Case

Ex. No	Estimated Probabilities from RF_L				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$			
14	0.002	0.008	0.063	0.927	3.915	0.106	11.766
15	0.005	0.128	0.113	0.754	3.616	0.523	10.673
16	0.080	0.155	0.609	0.156	2.841	0.606	8.188

Table E.3. Estimated Probability, the Expected Value, the Variance and SNR of RF_P for the Duplicator Case

Ex. No	Estimated Probabilities from RF_P				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y = I)$	$\hat{P}(Y = II)$	$\hat{P}(Y = III)$	$\hat{P}(Y = IV)$			
1	0.344	0.337	0.077	0.242	2.217	1.342	4.317
2	0.340	0.328	0.058	0.274	2.266	1.423	4.477
3	0.304	0.354	0.086	0.256	2.294	1.328	4.765
4	0.304	0.331	0.091	0.274	2.335	1.379	4.914
5	0.041	0.229	0.069	0.661	3.350	0.932	9.535
6	0.045	0.012	0.026	0.917	3.815	0.445	11.249
7	0.113	0.034	0.386	0.467	3.207	0.910	9.099
8	0.028	0.089	0.406	0.477	3.332	0.568	9.834
9	0.324	0.313	0.095	0.268	2.307	1.397	4.739
10	0.519	0.144	0.073	0.264	2.082	1.641	3.074
11	0.345	0.313	0.063	0.279	2.276	1.448	4.499
12	0.075	0.589	0.079	0.257	2.518	0.914	6.461
13	0.021	0.017	0.280	0.682	3.623	0.395	10.806

Table E.3 (cont'd). Estimated Probability, the Expected Value, the Variance and SNR of RF_P for the Duplicator Case

Ex. No	Estimated Probabilities from RF_P				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$			
14	0.008	0.010	0.005	0.977	3.951	0.115	11.840
15	0.028	0.082	0.062	0.828	3.690	0.546	10.847
16	0.188	0.109	0.462	0.241	2.756	1.043	7.308

Table E.4 Estimated Probability, the Expected Value, the Variance and SNR of RF_X for the Duplicator Case

Ex. No	Estimated Probabilities from RF_X				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y = I)$	$\hat{P}(Y = II)$	$\hat{P}(Y = III)$	$\hat{P}(Y = IV)$			
1	0.104	0.526	0.092	0.278	2.544	1.012	6.440
2	0.099	0.536	0.069	0.296	2.562	1.036	6.488
3	0.098	0.526	0.095	0.281	2.559	1.005	6.517
4	0.104	0.538	0.085	0.273	2.527	1.003	6.375
5	0.006	0.216	0.083	0.695	3.467	0.717	10.084
6	0.015	0.006	0.029	0.950	3.914	0.181	11.702
7	0.028	0.046	0.405	0.521	3.419	0.503	10.15
8	0.003	0.077	0.442	0.478	3.395	0.411	10.175
9	0.091	0.516	0.099	0.294	2.596	1.011	6.673
10	0.250	0.382	0.113	0.255	2.373	1.244	5.298
11	0.125	0.504	0.067	0.304	2.550	1.106	6.341
12	0.009	0.644	0.083	0.264	2.602	0.786	7.009
13	0.001	0.022	0.312	0.665	3.641	0.280	10.958

Table E.4 (cont'd). Estimated Probability, the Expected Value, the Variance and SNR of RF_X for the Duplicator Case

Ex. No	Estimated Probabilities from RF_X				$\hat{EV}$	$\hat{Var}$	$\hat{SNR}$
	$\hat{P}(Y=I)$	$\hat{P}(Y=II)$	$\hat{P}(Y=III)$	$\hat{P}(Y=IV)$			
14	0.003	0.007	0.012	0.978	3.965	0.066	11.911
15	0.004	0.075	0.056	0.865	3.782	0.345	11.252
16	0.068	0.143	0.551	0.238	2.959	0.652	8.549