Volume 9 | Number 4 | December 2023 | Pages 552-565

Jordan Journal of Electrical Engineering

ISSN (print): 2409-9600, ISSN (online): 2409-9619

Homepage: jjee.ttu.edu.jo

Bayesian Optimization Cost-Sensitive XGBoost Learning Algorithm for Imbalanced Data in Semiconductor Industry

Haziqah Shamsudin[1], Umi Kalsom Yusof[2*], Fizza Kashif[3], Iza Sazanita Isa[4]

[1, 2, 3, 4] School of Computer Sciences, Universiti Sains Malaysia, Pulau Pinang, Malaysia
E-mail: umiyusof@usm.my
[4] Centre for Electrical Engineering Studies, College of Engineering, Universiti Teknologi MARA, Pulau Pinang, Malaysia

Received: December 25, 2022 *Revised: March 20, 2023* *Accepted: April 01, 2023*

Abstract— This paper proposes an improved ensemble learning model based on extreme gradient boosting (XGBoost) with Bayesian optimization cost-sensitive learning algorithm for dealing with highly imbalanced data in the semiconductor process to achieve the highest possible pass and fail accuracy or recall for the classification performances. Most of the existing models are biased toward the majority class neglecting the minority class. The proposed Bayesian optimization cost-sensitive XGboost model is configured to be applied to the semiconductor dataset. The obtained experimental results - based on benchmarking semiconductor industry dataset - show 91.46% and 23.08% for the pass and fail accuracies, respectively. This confirms that the proposed model is significant for imbalanced cases in semiconductor applications. Moreover, this investigation reveals that the proposed model is able not only to maintain the performance of the majority class, but also to classify well the minority class.

Keywords— XGBoost learning algorithm; Cost-sensitivity; Imbalanced data; Semiconductor classification; Ensembled model.

1. INTRODUCTION

Semiconductor industry has proliferated since its inception and has a worth of 476.7 billion USD market all over the world with a yearly growth rate of 15.9% [1]. The demand for high-capacity and high-performance storage solutions such as hard disk drives and solid-state drives has been increasing rapidly and there is fierce competition among semiconductor industry players to stay ahead and control most of the market, i.e., the companies have to evolve and provide cutting edge technological solutions to meet the customers' demands. One of the key building blocks of these innovations is related to semiconductor manufacturing.

Semiconductor manufacturing is a highly complex manufacturing process composed of hundreds of steps [2], with numerous integrated circuits or "dies" are fabricated which diced from the wafer and put into the product. But, before the die is chosen, it must be classified based on its quality as depicted in Fig. 1. The dies are classified as "good" or "defective", the partial edge dies are discarded. The good dies are used in manufacturing high-end products like hard disk drives, whereas the defective dies are either used for simple products like pen

* Corresponding author Article's DOI: 10.5455/jjee.204-1671971895

drives or being discarded. However, semiconductor manufacturer is currently facing the problem of classifying dies quality because of class imbalance and the high complexity of the data. The classification algorithms favour the majority class and a wide majority of the dies are classified as good which means that many of the defective dies are also classified as good which is objectionable.

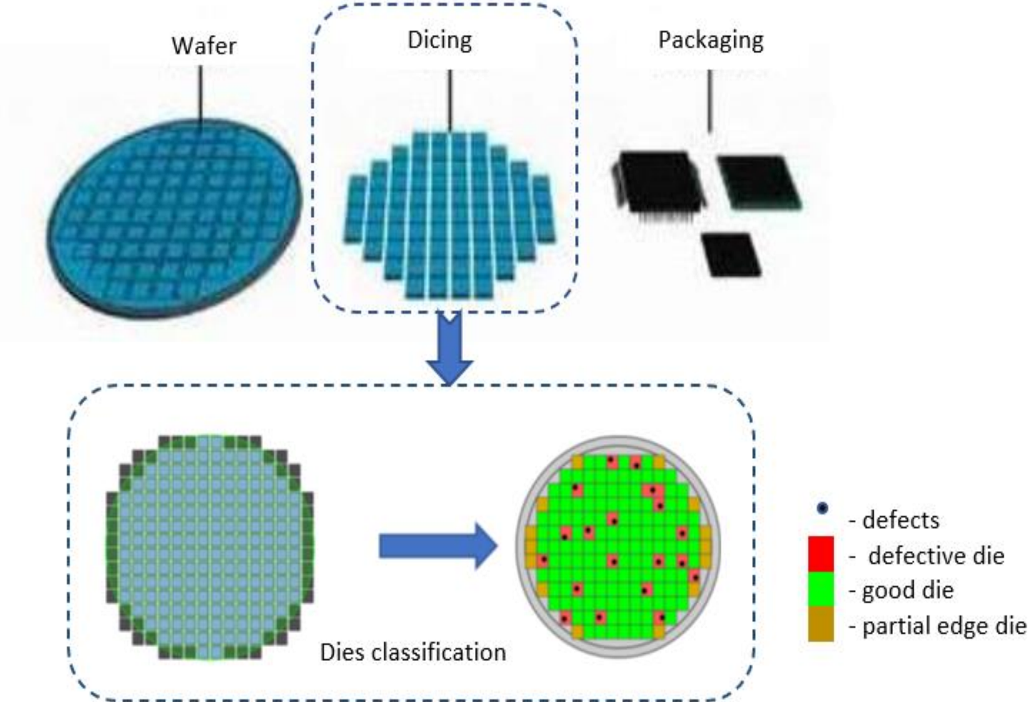

Fig. 1. Classification of dies quality in wafer fabrication.

The imbalanced learning problem is concerned with the performance of learning algorithms in the presence of underrepresented data, and severe class distribution skews [3]. Generally, the principal rule of machine learning algorithms assumes that the number of objects in considered classes is roughly similar [4]. However, in many real-life situations, the distribution of examples is skewed since representatives of some classes appear much more frequently. Hence, researchers started to propose new methods that try to tackle the imbalanced class problem but without considering the different complexity levels in the data. In this study, the main contributions are in two folds;

1. It focuses on the semiconductor domain of research by applying the SECOM industry benchmark dataset to maximize the classification quality of the algorithm.
2. By using the Bayesian optimization cost-sensitive XGBoost, the performance of the model shows an improvement compared to past literature.

There are several evaluation methods to measure the quality of a classification model. In addition to the existing evaluation methods, pass and fail accuracy are included as part of the evaluation measures. This strategy will enable the manufacturing company to ensure that product quality is not compromised under any circumstances.

The paper is organized as follows; in section 2, the background and related work are discussed. In section 3, the methodology and evaluation measures are discussed. In section 4, the results are presented and evaluated, and finally, conclusions are drawn in section 5.

2. BACKGROUND AND RELATED WORK

2.1. Dies' Classification in Semiconductor Industry

The classification of wafer dies in the semiconductor industry has been through extensive research and many solutions have been proposed for this classification. As example, Chen et. al [5] have proposed a two-stage approach; Least Absolute Shrinkage and Selection Operator (LASSO) detection and Random Forest algorithm with the embedded Status Variables Identification using sensors in wafers manufacturing to gather specific information about the wafer and die quality. However, this approach has focused primarily on operation selection and do not consider the class imbalance of the data and applicable only on specific machines.

Furthermore, Baly et. al [6] has proposed the Support Vector Machines algorithm to separate the good die from the bad by using algorithms like the C4.5 decision tree, k-NN, PLS regression, and GRNN, wherein their use of SVM generates results with a higher precision value. The limitation of this model is that the features must be reduced to achieve a higher level of precision that led to incur a loss of information, specifically when dealing with high dimensional data.

Meanwhile, an incremental clustering-based fault detection algorithm for class imbalanced process data proposed by Kwak et. al [7] has observed that this algorithm performs better than SVM and three instance-based fault detection algorithms. The study reported a higher accuracy with the incremental clustering-based fault detection algorithm but was limited by requirement to set the minority class to be exactly 20% for training and the synthetic test sets. Moreover, the data needed to follow a specified distribution, in which when dealing with real-world data, there is a slim chance the fault detection data will follow a purely elliptical or non-elliptical distribution like bivariate Gaussian or t-Copula.

An automated approach proposed by Imoto et. al [8] that used convolutional neural networks to identify the defective dies using three-phase classification model. In this study, the accuracy is observed to be higher as compared to the automatic defect classification (ADC) method. Some types of errors occur more often than others, for such defects the algorithm performs very well. However, the accuracy declines for the type of defects that occur rarely. This research has taken a very granular approach to the classification of the defective dies as it is done from the quality control perspective. The limitation of this model is that it requires a relatively large amount of accurately labelled input data for every defect. When the input data is not sufficiently available for each class of defect, the performance of the model will be affected accordingly, wherein the underrepresented class of defect will be misclassified.

2.2. Methods For Handling Imbalanced Data Classification

Handling semiconductor data can be challenging as it is typically comprising with more than 500 steps, and the amount of data recorded during the entire production process are vast amounts. In addition, most of the time, failures are rarely seen in this process. Being a highly expensive process, semiconductor industry has evolved to be inherently controlled on all front to lower the number of failures that occur in the process. This fact gives rise to

the challenge of imbalanced dataset, in which failure instances create a small portion of all observed instances [9].

Due to imbalanced class problem, classification of fabricated wafer quality in fabrication process has hardly decided in between pass and fail. In semiconductor industry, the classification process between pass and fail die must be extremely precise so as not to mislabel a die and end up with a substandard product. Any misclassification at this stage will incur a loss on the part of the company; either discarded the perfectly good die or employed defective die which will resulting in a substandard product. These errors lead to significant disadvantages to the semiconductor manufacturing company in terms of profits and customer satisfaction.

2.2.1. Under-Sampling and Over-Sampling Methods

In under-sampling methods, data is cut down and reduced so that all the classes are given equal representation [10] that completely changes in data dynamic and inordinate amount of information is lost which is not desirable when dealing with real-world data. Meanwhile, over-sampling methods increase the minority class instances artificially by generating synthetic data based on the existing real data. In this case, the data characteristics are lost and as the data was generated synthetically, the results will remain improbable. In addition to the loss of information, these methods tend to overfit the model towards the minority class and generate a higher generalization error. When the model is trained upon under or oversampled data, it will predict all the classes in almost equal amounts. When such a model is tested on real data, that will inherently be imbalanced, the performance metrics such as precision accuracy and recall will be affected consequently.

Oversampling on the other hand, is the most often used approach in dealing with the imbalanced class problem, as seen by the multitude of oversampling methods published in the last two decades. However, this does not necessarily imply that the oversampling approach is beneficial. Oversampling approaches boost the quantity of minority-class instances by creating new ones out of thin air based only on their similarity to one or more of the minority's examples. This is troublesome since such methods may raise the likelihood of the learning process being overfitted [11]. Another more critical problem of oversampling is that the fabricated examples could exist in the world belonging to a different class, regardless of how similar it is to the minority's examples, as we always have examples from class A that are the closest to examples from a different class B. Therefore, we argue that, even if such synthesizing generates favourable outcomes on paper, negative results can be easily obtained in practice.

Oversampling methods consumed more time for modelling as there is an increase in the amount of data, whereas under-sampling is computationally expensive as the data pre-processing is extensive. In general, Rekha et. al [12] has observed that under-sampling provides higher precision in her study, but it was coming at the cost of loss of information when many of the instances are removed to balance the data. When dealing with real-world data, where decisions must be made that will impact significant aspects of a business, these results might not be applicable or acceptable.

2.2.2. Cost Sensitive Learning

Cost-sensitive learning can fill this gap in the modelling process, where no information is lost, and all relevant data is used to train the model for future cases of classification by assigning appropriate cost values to erroneous results. Using cost-sensitive learning is a better alternative to over and under-sampling the data. Defining cost measures to train the models is the critical part of cost-sensitive learning. Lower cost measures are unable to adjust the decision boundaries effectively, contrarily the higher cost measures lead to poor generalization capacity on non-penalized classes [13]. The particularities of costs are largely depending on the domain should be validated by experts. Cost-sensitive learning has been employed to tackle the class imbalance problem by numerous researchers in varied fields. But cost-sensitive learning alone is not the complete solution in most cases. Conjunction with pre-processing methods like feature selection and sophisticated modelling methods like ensemble and hybrid algorithms can be applied.

To date, only few researchers are investigating to resolve issues of highly dimensional and imbalanced data in semiconductor industry while maintaining the original data integrity to ensure that all characteristics of the data are used to train the model and no generalization is made. However, more computational capacity in classifying highly imbalanced data and cost sensitive algorithm processing is needed to resolve these issues. Therefore, this study aimed to develop a new optimized model using cost sensitive learning algorithm for classification in which the trained model is penalized with a predefined cost parameter, which enables the classifier to take the potential loss associated with a wrong decision into consideration.

2.2.3. Machine Learning-Based Methods for Imbalanced Data Classification

Recently, machine learning offered a wide range of algorithms to be employed in classification tasks where unseen instances must be put into a predefined category based on data attributes. These algorithms include but are not limited to decision trees, k-nearest neighbor, logistic regression, support vector machines, and naïve Bayes. Standard classification algorithms based on data with an imbalanced distribution will yield biased results in the favour of the majority class, whereas the minority class will be frequently misclassified. Class imbalance can be handled with a few techniques, i.e., over-sampling the underrepresented class, under-sampling the over-represented class, or using a cost function within the model, that will penalize the misclassification errors: cost-sensitive learning.

Ensemble learning is a powerful technique in machine learning wherein two or more base learners are conjoined together to yield a better result than that generated by a single model [14]. Machine learning models are many and the combination of two or more of these models allows researchers endless possibilities to enhance their performance metrics. Ensemble learning methods are mainly divided into sub-categories like voting, bagging, boosting, and stacking, etc. [15]. The most used methods in ensemble algorithms are Random Forest (RF) and XGBoost [16].

RF works by combining multiple decision trees, each of which is trained on a randomly selected subset of the training data and a random subset of the features. During training, the algorithm searches for the best feature to split the data at each node, based on a measure of purity such as Gini impurity or information gain. Once the trees have been constructed,

predictions are made by aggregating the predictions of each individual tree. In classification problems, the final prediction is the class that receives the most votes from the individual trees [17].

While RF can be a good model, some limitations of the algorithm such as it's prone to overfitting when dealing with noisy data or datasets with high number of features [17]. In addition, RF can be slower in computation time when dealing with large datasets, especially if the number of features is high. This is because RF constructs multiple decision trees in parallel.

On the other hand, XGBoost is an ensemble model in which decision trees are made for the classification task sequentially. Every data point is assigned a weight value, through which a probability is calculated which leads to the final decision of node allocation. Initially, all data points are given the same weights and these weights are then adjusted on every iteration where the model retains the decisions made on the previous weights and adjusts the weights in the next step based on the analysis. This process is repeated until the final classifier is built for classification [18].

An extreme gradient boosting algorithm or XGBoost is one of the most widely used Gradient Boosted Machine (GBM) in the field of industry and manufacturing because of its high performance in problem-solving and marginal need for feature engineering [19]. Since the data generated in the semiconductor industry is highly complex, further feature engineering makes the task even more complicated. The methods that eliminate the chance of confounding the data even further are more appropriate. The strengths of this classifier are considered carefully which has led to this algorithm being selected as one of the models to test on the dataset.

2.3. Hyperparameter Optimization

The utilization of machine learning algorithms has been extensive across various domains and fields. For a machine learning model to be applied to different problems, it is essential to fine-tune its hyperparameters. The performance of the model is directly impacted by the selection of the best hyperparameter configuration. To create an optimal ML model, a range of potential options must be explored. The process of designing an ideal model structure with the best hyper-parameter setup is referred to as hyperparameter tuning. Different techniques can be employed for hyperparameter optimization, including grid search (GS), random search (RS), Bayesian optimization (BO), and other approaches [20].

One of the popular techniques to investigate hyperparameter configuration space is known as GS [21]. This method can be considered as an exhaustive or brute-force search, which involves assessing all the possible combinations of hyperparameters on a given grid. The approach involves evaluating the cartesian product of a finite set of values specified by the user. Although GS can be easily implemented and parallelized, it suffers from a major limitation for high-dimensional hyperparameter configuration spaces. As the number of hyperparameters increases, the number of evaluations required increases exponentially, making GS inefficient. Therefore, it is recommended to use GS only when the hyperparameter configuration space is relatively small [20].

In contrast to GS, RS was introduced in [22]. This technique is similar to GS, but it randomly selects a set number of samples between the upper and lower bounds as

hyperparameter candidates and trains them until a given budget is consumed, instead of testing all the values in the search space. The underlying principle of RS is that if the configuration space is extensive enough, it is possible to detect the global optimum or at least approximate it. Although RS is more efficient than GS for vast search spaces, there may still be a significant number of unnecessary function evaluations since RS does not take advantage of the previously well-performing areas [20].

To conclude, the main limitation of both RS and GS is that every evaluation in their iterations is independent of previous evaluations; thus, they waste massive time evaluating poorly performing areas of the search space. This issue can be solved by other optimization methods, like Bayesian optimization that uses previous evaluation records to determine the next evaluation.

Bayesian optimization (BO) is a well-known algorithm for solving HPO problems [23]. Unlike GS and RS, BO selects future evaluation points based on past results. It uses two main components, a surrogate model and an acquisition function, to determine the next hyper-parameter configuration [24]. The surrogate model is responsible for fitting all observed points into the objective function, while the acquisition function balances exploration and exploitation to choose the best points to evaluate. Exploration involves sampling instances from unexplored regions, while exploitation involves sampling from regions that are likely to contain the global optimum based on the predictive distribution of the surrogate model. BO models balance exploration and exploitation to identify the most promising regions and avoid missing better configurations in unexplored areas [20].

Considering the related works in the domain of classification in the semiconductor industry and class imbalance problems, there are a few methods that can be applied to solve specific problems for the manufacturing process. As established, under-sampling and over-sampling have their limitations in terms of information loss and synthetic information that may not reflect the true characteristics of the data. These gaps can be filled with a cost-sensitive machine learning model that employs all the information in the original data without changing it to balance the class labels. At the same time, to optimize well the cost value, Bayesian optimization is relevant to be integrated due to the capability of the model in improving the parameter optimization.

3. METHODOLOGY

The research framework of this study is designed to propose a method for optimizing the Cost-Sensitive XGBoost model. The data used for this study is a public data from the UCI machine learning repository (SECOM) as semiconductor industry data [5]. Generally, the work starts with preprocessing and cleaning the datasets into an appropriate format of consistencies to gain a valid result. This step usually requires the maximum amount of time in the data science process. As seen in the research framework as illustrated in Fig. 2, there are different pipelines for the data discussed above. Next, the ensemble model of the cost-sensitive XGBoost algorithm is developed based on the data and the selected model is tested. Finally, the performance of the proposed model is optimized and evaluated. The experiments conducted in this project were run on an Intel® Core™ i7 CPU 2.60 GHz with 8.00 GB of RAM under a 64-bit Windows 10 Enterprise operating system.

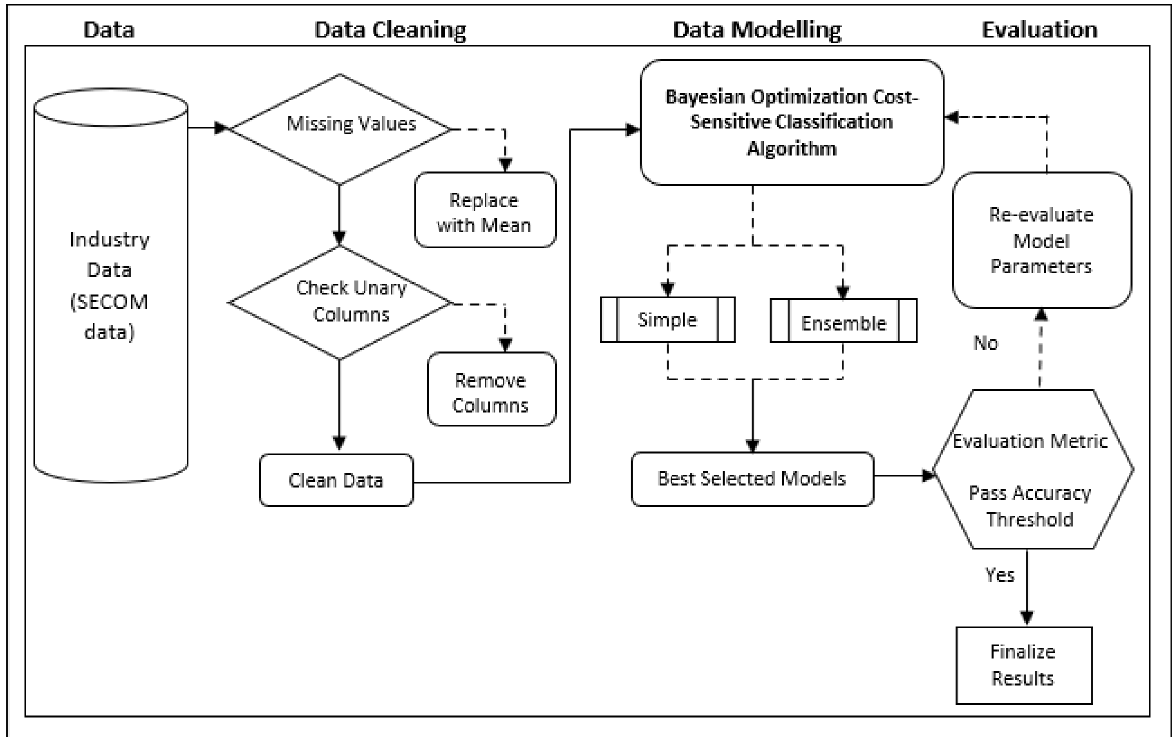

Fig. 2. The proposed method for optimizing the cost-sensitive XGBoost model.

3.1. Data Preprocessing and Cleaning

The SECOM is a public dataset available on the UCI machine learning repository and Kaggle which donated to the repository in 2008 that containing sensor data collected during the manufacturing process in the semiconductor industry. The specifics of the data are detailed in Table 1. As can observe that the number of features is significantly high which translates to higher data complexity when designing a modelling task. Many of these features will be redundant on the target variable so data preprocessing and cleaning is vital to achieving reliable results. The target label in this dataset is a categorical variable which records whether a die is defective or not. The number of defective dies is only 104 as opposed to the good dies which is 1463. Herein lies the class imbalance of the data. In this paper, the degree of imbalance of data will be determined by the imbalance ratio or IR, which is defined as the ratio of instances in majority class to those in the minority class. Data for which the IR value is 9 is considered as highly imbalanced and should be subjected to appropriate methods rather than the traditional algorithms which are designed to treat all classes as equal [25].

Table 1. Configurations of the SECOM dataset.

Data Characteristics	Number
Observations	1567
Features	591
Imbalance Ratio	14

In the case of the benchmark dataset, the number of features is 591, and the data has many missing values. If the rows with missing values are dropped, the output is a blank vector, which implies that all the rows of the data have at least one missing observation. So, the only option available was to replace the missing values with an appropriate measure.

There are options like replacing with mean, median, or simply 0. By closely observing the data, the decision was made to replace all missing with the mean of the respective column. Since there were a significant number of missing values, replacing them with zero would have compromised the integrity and distribution of the data. Additionally, the data does not have any extreme outliers, otherwise, the median would have been a better option.

The unary columns were dealt after treating the missing values. A column that records the same value in all the rows is known as a unary column and does not contribute to the outcome of the class label since it remains constant throughout the process. These columns were identified and completely dropped from the data as they are not contributing in any way. The timestamp was also dropped as it was not relevant to the target label. Feature selection was skipped for this data since the features to be used in the real-semiconductor data are predetermined by the engineering department of the company so, to retain the maximum information, the remaining columns were maintained for the modeling phase.

3.2. Cost-Sensitive XGBoost Model

In the modelling phase, the algorithms were divided into single and ensemble models. The inclination was maintained to switch to the ensemble model if and only if it gives a better score than the simple model. Since opting for a more complicated model is not justified if the results are not improved. The computational cost needs to be justified while proposing an ensemble model.

3.2.1. Cost-Sensitive Learning

In cost-sensitive learning, the cost of false positive (actual negative but predicted as positive; denoted as FP), false negative (FN), true positive (TP) and true negative (TN) can be given in a cost matrix, as shown in Table 2. The notation $C(i, j)$ is being used to represent the misclassification cost of classifying an instance from its actual class j into the predicted class i. In cost-sensitive learning, it is usually assumed that such a cost matrix is given and known.

Table 2. An example of cost matrix for binary classification

	Actual Negative	Actual Positive
Predict Negative	C (0,0) or TN	C (0,1) or FN
Predict Positive	C (1,0) or FP	C (1,1) or TP

When dealing with the real-semiconductor data, the challenge was to determine the cost measure that would reflect the data characteristics. Since no monetary value could be retrieved to make a cost matrix, the task came down to tuning the hyperparameter. The default value of this parameter is 1: all classes are assumed to have equal importance. The value of this parameter can be set to any number, but it cannot be any random number. It must depict the qualities of the data used for training. When observing the training data, the class imbalance was quite significant as it is evident from the IR value of 13.

3.2.2. XGBoost Algorithm

The XGBoost algorithm is employed in this study because of its high performance in problem-solving and minimal need for feature engineering. This quality of XGBoost makes it

a viable candidate for the case because, besides feature selection, no further modifications have been made to the original data. The XGBoost relies on the sequential decision trees where a weight is assigned to every data point and the next round is evaluated by keeping the weights in the previous step in consideration. The overall process of the ensemble model is shown in Fig. 3.

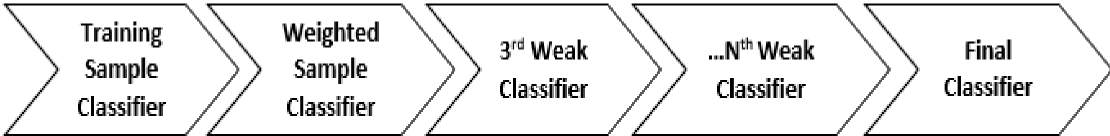

Fig. 3. The XGBoost algorithm used in the ensemble model.

Mathematically, if m is number of features and n number of examples as in Eq. (1),

$$D=\{(x_i,y_i)\}\ (|D|=n, x_i\in R^m, y_i\in R) \quad (1)$$

where D is an ensemble model of a tree that uses K additive functions to predict the output, as shown in Eq. (2), given that F is the feature space,

$$y_i=\varnothing(x_i)=\sum_{k=1}^{K} f_k(x_i), f_k\in F \quad (2)$$

where $F=\{f(x)=w_{q(x)}\}(q: R^m \rightarrow T, w \in R^T)$ is the space of regression trees (also known as CART). Here q represents the structure of each tree that maps an example to the corresponding leaf index. T is the number of leaves in the tree. Each f_k corresponds to an independent tree structure q and leaf weights w [18].

The XGBoost algorithm has numerous hyperparameters that need to be tuned so that the algorithm can be employed to its full potential. Specifically, for imbalanced datasets the hyperparameter "scale pos weight" allows the user to custom-define the weights for the classes in the dataset. Here, the minority class can be given higher importance as opposed to the majority class so that the algorithm penalizes the misclassification errors accordingly. This hyperparameter is employed in the deployment of the final model to penalize the errors and increase the overall pass accuracy.

In this paper, the parameter value for scale pos weight is taken from the value of the imbalance ratio a shown in Eq. (3), which defined N^+ and N^- as majority and minority class sample size, respectively.

$$Scale\ pos\ weight = \frac{N^+}{N^-} \quad (3)$$

The minority class can be given higher importance as opposed to the majority class so that the algorithm penalizes the misclassification errors accordingly. This hyperparameter is employed in the deployment of the final model to penalize the errors and increase the overall pass accuracy.

3.3. Performance Evaluation

Evaluating algorithms for classification tasks, the traditional metrics are generated from the confusion matrix. These metrics include accuracy, precision, recall, and F1- Score. The confusion matrix summarizes the actual and the predicted outcomes in a tabular format

in terms of FP, FN, TP, and TN. Ideally, the off diagonals i.e., the FP and FN results should be zero [12]. The traditional measures are precision and recall which are defined as, fraction of the predicted positive values that are indeed positive and fraction of positives that were correctly predicted by the classifier, respectively. The threshold or the F1- Score (Eq. (4)) is the measure used to define a balance between precision (Eq. (5)) and recall (Eq. (6)) [18]. To check the individual performance of the model against the class label, pass accuracy and fail accuracy were used in this paper. The individual accuracy is checked to understand the behavior of the model towards the individual class. The pass accuracy is also equivalent to the recall formula.

$$F1-Score = \frac{2(Precision)(Recall)}{Precision + Recall} \quad (4)$$

$$Precision = \frac{TP}{(TP + FP)} \quad (5)$$

$$Recall = \frac{TP}{(TP + FN)} \quad (6)$$

$$Fail\ Accuracy = \frac{TN}{(TN+FP)} \quad (7)$$

The traditional methods may be effective but only to a limited extent when it comes to highly imbalanced data. The overall accuracy cannot be relied on to be an appropriate evaluation metric, and neither can the error rate. In imbalanced data, precision and recall can be observed together with the F1-Score to make a sound final decision on the model performance. Rather, the additional measure that is used to judge a model is the pass accuracy (Eq. (6)) and fail accuracy (Eq. (7)) which is defined as the ratio of correctly classified good dies with the total good dies and vice versa. Since, in such highly imbalanced data, every measure cannot be optimized so the strategy is to prioritize the measure with the most impact in the final manufacturing process.

4. RESULTS AND DISCUSSION

4.1. Comparison Results of Simple and Ensemble Models

The simple model of Decision Tree (DT) was compared with ensemble models of XGBoost on benchmark dataset. The comparison results are showing in Table 3. As we can see that the evaluation metric of interest, which is the pass accuracy has improved when using an ensemble model. Even though DT is able to classify better the fail class, but in terms of the equality in performance, XGBoost baseline model is able to perform better. As can be observed in Table 3, the overall performance of precision, F1-Score and Recall shows a positive increment compared to DT which indicates XGBoost have the potential to classify better even at the presence of imbalanced class problem.

Table 3. Comparison results of simple and ensemble models on SECOM dataset.

Model	True	Predicted		Accuracy [%]		Precision [%]	F1-Score [%]	Recall [%]
		0	1					
Decision Tree	0	3	23	Fail	11.54	0.06	0.03	0.03
	1	47	398	Pass	89.44	0.95	0.92	0.94
XGBoost	0	1	25	Fail	3.85	0.33	0.07	0.04
	1	2	443	Pass	99.55	0.95	0.97	1.00

4.2. Comparison Results of Default XGBoost, Cost XGBoost, and Bayesian Optimization Cost-Sensitive XGBoost on SECOM Dataset

The results for the default XGBoost and the cost-sensitive XGBoost are shown in Table 4. The failure class is represented by 0 and the passed class by 1. The benchmark data size is significantly smaller than the real-semiconductor data, so any small improvement is considered as a big leap forward because it translates well when applied to the larger dataset.

From Table 4, adding cost value improved the performance of the model in classifying both Pass and Fail classes. However, due to limitation in deciding the parameter of the cost, integration with Bayesian optimization model is needed to enhance the model. Bayesian optimization is used to improve the parameter configuration of the cost XGBoost model. It shows significant improvement in not only the pass accuracy, but at the same time the fail accuracy, precision, F1-score and recall. Despite the pass accuracy shows a small drop, but it is still reliable since the performance of the Bayesian Cost XGBoost is still high.

Table 4. Comparison results of default XGBoost, Cost-Sensitive XGBoost and Bayesian Optimization Cost-Sensitive XGBoost models on SECOM dataset.

Model	True	Predicted		Accuracy [%]		Precision [%]	F1-Score [%]	Recall [%]
		0	1					
XGBoost	0	1	25	Fail	3.85	0.33	0.07	0.04
	1	2	443	Pass	99.55	0.95	0.97	1.00
Cost XGBoost	0	2	24	Fail	7.69	0.25	0.12	0.08
	1	6	439	Pass	98.65	0.95	0.97	0.99
Bayesian Optimization Cost-Sensitive XGBoost	0	6	20	Fail	23.08	0.14	0.17	0.23
	1	38	407	Pass	91.46	0.95	0.93	0.91

When this model is deployed in the manufacturing process at industry, it will benefit the company in terms of the confidence it entails in the decision about the good dies. Technically, the company can test a new wafer and when a die is classified as “good”, there will be highly confidence of 91.46% in the decision. The proposed XGBoost model with added bayesian cost parameter is significant for implementation in the recent industrial application. As a summary, the state-of-the-art for proposed model compared to other methods is presented in Table 5.

Table 5. Comparisons of the traditional algorithm (literature) with default XGBoost, Cost XGBoost, and proposed Bayesian Optimization Cost-Sensitive XGBoost on SECOM dataset.

Method	Pass Accuracy [%]	Fail Accuracy [%]
Naïve Bayes [2, 25]	74.60	64.79
XGBoost	99.55	3.85
Cost XGBoost	98.65	7.69
Bayesian Optimization Cost-Sensitive XGBoost (Proposed)	91.46	23.08

5. CONCLUSIONS AND RECOMMENDATION

Many semiconductor manufacturing companies are striving to improve the performance of their classification algorithms when classifying the goods and the defective

dies for the manufacturing process. The good die significantly outweighs the defective ones in most cases.

To solve this problem, a Bayesian optimization cost-sensitive extreme gradient boosting approach is proposed in this paper. The classes are balanced by using the Bayesian optimization of the training data as a cost hyperparameter within the XGBoost model. Keeping the highly imbalanced nature of the data, for practical reasons, all evaluation metrics are focused on each class's accuracy, precision, recall, and F1-score. By using cost-sensitive learning, these values have been maximized.

Some limitations in this study included the high number of features and high complexity of the data which cause difficulties when tuning the model to find the best model. Though it shows an improvement, potentially in the future, more tuning can be done and at the same time, integrating XGBoost with other models will be helpful in improving further the overall performance.

In continuation of the research carried out in this study, future works can involve experimentation with an actual cost matrix provided by a domain expert that accurately portrays the losses associated with the misclassification. The results of such works can be evaluated in terms of monetary value, for example, the cost is reduced by a specified amount by using cost-sensitive learning. In contrast, the cost matrix can be defined in terms of gain; how much a company can gain if the element is classified correctly.

REFERENCES

[1] A. Ahmad, "Automotive semiconductor industry-trends, safety and security challenges," *Proceedings of 8th IEEE International Conference on Reliability, Infocom Technologies and Optimization*, pp. 1373-1377, 2020.

[2] M. Sathyan, R. Balakrishnan, "Predictive models for equipmentfault detection in the semiconductor manufacturing process," *IACSIT International Journal of Engineering and Technology*, vol. 8, no. 4, pp. 273-285, 2016.

[3] H. Ali, M. Salleh, R. Saedudin, K. Hussain, M. Mushtaq, "Imbalance class problems in data mining: A review," *Indonesian Journal of Electrical Engineering and Computer Science,* vol. 14, no. 3, pp. 1552-1563, 2019.

[4] W. Zheng, M. Jin, "The effects of class imbalance and training data size on classifier learning: an empirical study," *SN Computer Science,* vol. 1, no. 2, pp. 1-13, 2020.

[5] Y. Chen, "Big data analytic for multivariate fault detection and classification in semiconductor manufacturing," *Proceedings of IEEE Conference on Automation Science and Engineering*, pp. 731-736, 2017.

[6] R. Baly, H. Hajj, "Wafer classification using support vector machines," *IEEE Transactions on Semiconductor Manufacturing* , vol. 25, no. 3, pp. 373-383, 2012.

[7] J. Kwak, "An incremental clustering-based fault detection algorithm for class-imbalanced process data," *IEEE Transactions on Semiconductor Manufacturing* , vol. 28, no. 3, pp. 318-328, 2015.

[8] K. Imoto, T. Nakai, T. Ike, K. Haruki, Y. Sato, "A CNN-based transfer learning method for defect classification in semiconductor manufacturing," *IEEE Transactions on Semiconductor Manufacturing*, vol. 32, no. 4, pp. 455-459, 2019.

[9] S. Milad, T. Shayan, Y. Jiann-Shiun, "An experimental evaluation of fault diagnosis from imbalanced and incomplete data for smart semiconductor manufacturing," *Big Data and Cognitive Computing*, vol. 2, no. 4, pp. 30, 2018.

[10] J. Du, "Online sequential extreme learning machine with under-sampling and over-sampling for imbalanced big data classification," *Proceedings of ELM Springer*, pp. 229-239, 2018.

[11] A. Tarawneh, A. Hassanat, G. Altarawneh, A. Almuhaimeed, "Stop oversampling for class imbalance learning: a review," *IEEE Access*, vol. 10, pp. 47643-47660, 2022.

[12] G. Rekhaa, A. Tyagi, V. Krishna Reddy, "Performance analysis of under-sampling and over-sampling techniques for solving class imbalance problem," *Proceeding of International Conference on Sustainable Computing in Science, Technology and Management*, pp. 1305-1315, 2019

[13] Q. Xu, S. Lu, W. Jia, C. Jiang, "Imbalanced fault diagnosis of rotating machinery via multi-domain feature extraction and cost-sensitive learning," *Journal of Intelligent Manufacturing*, vol. 31, pp. 1467-1481, 2020.

[14] G. Xu, M. Liu, Z. Jiang, D. Soffker, W. Shen, "Bearing fault diagnosis method based on deep convolutional neural network and random forest ensemble learning," *Sensors*, vol. 19, no. 5, pp. 1088, 2019.

[15] E. Alpaydin, *Introduction to Machine Learning*, MIT Press, 2020.

[16] E. Sahin, "Assessing the predictive capability of ensemble tree methods for landslide susceptibility mapping using XGBoost, gradient boosting machine, and random forest," *SN Applied Sciences,* vol. 2, no. 7, pp. 1308, 2020.

[17] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5-32, 2001.

[18] T. Chen, C. Guestrin, "Xgboost: a scalable tree boosting system," *Proceedings of the 22nd ACM Sigkdd International Conference on Knowledge Discovery and Data Mining*, pp. 785-794, 2016.

[19] J. Duan, P. Asteris, H. Nguyen, X. Bui, M. Hossein, "A novel artificial intelligence technique to predict the compressive strength of recycled aggregate concrete using ICA-XGBoost model," *Engineering with Computers*, vol. 37, pp. 3329-3346, 2021.

[20] L. Yang, A. Shami, "On hyperparameter optimization of machine learning algorithms: theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, 2020.

[21] M. Injadat, A. Moubayed, A. Nassif, A. Shami, "Multi-split optimized bagging ensemble model selection for multi-class educational data mining," *Applied Intelligence*, vol. 50, pp. 4506-4528, 2020.

[22] J. Bergstra, Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. 2, pp. 281-305, 2012.

[23] J. Snoek, H. Larochelle, R. Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1-9, 2012.

[24] M. Injadat, F. Salo, A. Nassif, A. Essex, A. Shami, "Bayesian optimization with machine learning algorithms towards anomaly detection," *In 2018 IEEE Global Communications Conference*, pp. 1-6, 2018.

[25] J. Kim, Y. Han, J. Lee, "Data imbalance problem solving for smote based oversampling: study on fault detection prediction model in semiconductor manufacturing process," *Advanced Science and Technology Letters*, vol. 133, pp. 79-84, 2016.