International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Silicon Wafer Fault Detection Using Machine Learning Techniques

B P Swathi*[1], Divya Shree K V [2], Aaditya Balakrishna [3], Harshitha Jampala[4], Geetishree Mishra[5]

Submitted: 26/01/2024 **Revised**: 04/03/2024 **Accepted**: 12/03/2024

Abstract: In recent years, the growing demand for electronic devices has underscored the critical role of semiconductor manufacturing. This industry focuses on transforming semiconductor materials into integrated circuits, demanding precision and utilizes advanced technologies to etch intricate patterns onto silicon wafers. However, the acquisition of comprehensive datasets pertaining to wafer production encounters formidable challenges such as data imbalance and noise. This research explores the application of machine learning techniques and deep neural networks for the classification of defective wafers, assessing their impact on managing semiconductor manufacturing processes using the Semiconductor Manufacturing Process (SECOM) dataset. Subsequently, various methodologies, including sample-based, instance-based, ensemble learning, and Support Vector Machine approaches, were implemented and rigorously evaluated to provide a thorough comparison.

Keywords: *Data preprocessing, Ensemble learning, Machine learning, Neural networks, Semiconductor manufacturing.*

1. Introduction

The semiconductor industry is undergoing rapid expansion, driven by the widespread integration of sensor based and automated technologies, particularly with the adoption of smart components and the Internet of Things (IoT) in various domains. The semiconductor manufacturing process involves various stages, including wafer fabrication, assembly, packaging, and final integrated circuit testing. While the majority of the steps are automated and executed in a highly controlled, ultraclean environment to minimize errors and defects, occasional fabrication errors may still arise at different stages of the manufacturing process. The process of wafer fabrication involves photolithographic patterning that can cause incorrect dimensions and out of specification performance. Etching, wafer doping, and the deposition of metal can alter the electronic levels and fluctuations in temperature, pressure, and timing of the semiconductor. These errors have the potential to degrade the performance and reliability of semiconductor devices.

[1] *Department of Electronics and Communication Engineering, B.M.S. College of Engineering, Bengaluru, Karnataka – 560019, INDIA*
Email: swathi.ec20@bmsce.ac.in
ORCID ID : 0009-0006-4598-6071
**(Corresponding Author)*
[2] *Department of Electronics and Communication Engineering, B.M.S. College of Engineering, Bengaluru, Karnataka – 560019, INDIA*
Email: divya.ec20@bmsce.ac.in
ORCID ID : 0009-0002-8482-9387
[3] *Department of Electronics and Communication Engineering, B.M.S. College of Engineering, Bengaluru, Karnataka – 560019, INDIA*
Email: aaditya.ec20@bmsce.ac.in
ORCID ID : 0009-0006-0521-0316
[4] *Department of Electronics and Communication Engineering, B.M.S. College of Engineering, Bengaluru, Karnataka – 560019, INDIA*
Email: harshithaj.ec20@bmsce.ac.in
ORCID ID : 0009-0006-5820-741X
[5] *Assistant Professor, Department of Electronics and Communication Engineering, B.M.S. College of Engineering, Bengaluru, Karnataka – 560019, INDIA*
Email: geetishreemishra.ece@bmsce.ac.in
ORCID ID : 0000-0002-7781-7794

The complexity of analyzing manufacturing processes is further compounded by the integration of numerous sensors on production line equipment, generating substantial data volumes. Increased sampling rates result in the need for big data and machine learning technologies to effectively manage manufacturing processes. Analyzing sensor data provides benefits such as early problem detection, enhanced defect detection, improved quality, and waste reduction.

This paper addresses significant challenges in manufacturing process management, particularly focusing on high-dimensional data and class imbalance. Other challenges include dataset shift, incremental learning, noisy data, and budget constraints. The curse of dimensionality is associated with reduced generalization ability in classification algorithms. We introduce a novel preprocessing approach to address these issues and apply various classification models, such as K–Nearest Neighbor, Decision Tree, Support Vector Machine, Logistic Regression, and Naive Bayes, along with ensemble methodologies like Random Forest. Additionally, Deep neural networks are employed to analyze the model adeptness in handling complexes.

The Literature Survey section provides an overview of the pertinent background and prior work. The Methodology and Implementation section delves into the exploration of various models and their application to the problem under consideration. The Results section discusses the outcomes of the various models and the inferences drawn. The paper culminates with a brief discussion on the performance of the models and their potential applicability in industrial manufacturing processes, emphasizing the significance of overcoming the outlined challenges.

2. Literature Survey

In the realm of semiconductor manufacturing processes and classification, notable contributions have been made in recent studies. One study [1] delved into predictive models for equipment fault detection, employing a comprehensive methodology which involved steps such as data preparation, data cleansing, feature scaling, feature reduction, feature selection, variable component analysis, and model selection. Concurrently, another study [2] investigated the impact of intrinsic dataset characteristics on classification performance, utilizing techniques such as Principal Component Analysis (PCA) and brute force (w-SimpleCART).

A Deep Neural Network (DNN) model to address wafer map imbalance was suggested by J. Wang et al. [3]. However, concerns about DNN's suitability for yield prediction have been raised, citing issues like overfitting and poor model visibility [12]. The current study aims to make a significant contribution by addressing the gap in Final Time (FT) yield prediction using Wafer Acceptance Testing (WAT) data at the initial Wafer Fabrication (WF) stage. Effectively overcoming the challenges posed by high-dimensional input data and complex process variations necessitates a careful selection of machine learning techniques [4].

An exploration on equipment-related variable selection for WS yield was made by Kim et al. [5] Their model predicted the performance of two wafer process parameters but lacked visibility into the overall yield. Moreover, it did not address the relationship between the two measurements. Few studies delve into FT yield prediction; S. Kang et al. [5] used WS measurements and wafer spatial features for die-level FT yield prediction. No prior study focuses on FT yield prediction using WAT data. Past studies often predict specific failure modes, lacking coverage for all types of failures. The input data, including wafer die features, Wafer Sort (WS) measurements, inline metrology data, or process equipment information, are specific to each stage, restricting their scope.

These features were employed by Kim et al. [6], with a focus on yield issues related to the lithography process and utilizing the DNN algorithm. Nakata et al. [8] utilized Convolution Neural Network (CNN) for monitoring wafer map failure patterns, demonstrating superior performance compared to SVM. Significant contributions from Kang et al. [9] and Park et al. [14] delved into backend yield related issues, predicting FT yield using probe test parametric data and introducing a FT yield classifier based on wafer probe test results and wafer map features.

A model for predicting die-level yield using spatial features of wafer dies was introduced by Jang et al. [11]. Kong and Ni [12] utilized SVM, assuming that wafer yield loss is primarily driven by inline defects, potentially limiting its applicability. Prior studies [13] have addressed three major challenges in analyzing data from hundreds of signals or sensors associated with the SECOM [21] dataset. Firstly, the issue of feature selection involves overcoming obstacles such as a large number of features, noise, and irrelevant information. Secondly, addressing the imbalance between pass and fail cases in each measurement point is crucial. Thirdly, determining the most suitable classification algorithm for effective analysis.

Various approaches have been explored in existing studies. For example, in one study [15], feature selection methods, including the removal of constant and missing value features, Chi-Square statistical analysis, PCA, and Gain Ratio, were employed. Tested algorithms encompassed k-Nearest Neighbor (KNN), Logistic Regression (LR), Naïve Bayes (NB), and Decision Trees (DT). Evaluation metrics included True Positive Rate (TPR), Precision, F-measure, and False Positive Rate (FPR), with the Waikato Environment for Knowledge Analysis (WEKA) software used for feature selection.

Past investigations in the semiconductor sector mainly focused on applying machine learning techniques to front-end processes, encompassing enhancements in virtual metrology, fault detection and classification, wafer yield estimation, and probe yield analysis [16]. Research on semiconductor yield prediction primarily concentrates on WF and WS stages [17], aiming at optimizing wafer map design, monitoring defect patterns, and predicting wafer yield. For example, wafer map design is crucial for addressing wafer low yield problems [18]. The semiconductor manufacturing yield prediction faces two major challenges—high-dimensional input data and intricate process variations—leading to the exploration of diverse approaches. Methods for feature reduction and selection, such as Pearson correlation, Mutual Information (MI), Recursive Feature Elimination (RFE), and Hybrid Feature Selection (HFS), have been investigated to address high-dimensional input data [3][10][22][24]. In tackling complex process variations, diverse techniques have been employed, encompassing multilevel lasso models, PCA, and regression-based models [14][24].

Addressing data imbalance in yet another study [19], the Synthetic Minority Oversampling Technique (SMOTE) was proposed. This technique oversampled the fail class to resolve the imbalance, with a comparison of Random Forest (RF), Artificial Neural Network (ANN), LR, and DT algorithms. Dealing with incomplete data, a different study [20] concentrated on missing value replacement using methods derived from DT, NB and Nearest Neighbor (NN). Common substitution approaches included using the mean or mode. Investigating class imbalance, class overlap, and lack of density. A noteworthy application of PCA by Chen [22], enhancing the forecasting performance of the Fuzzy Back Propagation Network (FBPN) for WF cycle time

estimation, stands out. However, utilizing PCA for feature selection in FT yield prediction may lead to information loss and diminished interpretability.

An effective feature selection approach based on HFS, eliminating noise parameters to enhance the accuracy of wafer probe yield prediction was suggested by Xu et al. [22]. The primary concept of clustering involves grouping models into several clusters and selecting representative models (one or more) from each cluster [23]. In this study, they employ the cluster ensemble method, combining multiple partitioning of a set of objects without accessing the original features. The objective is not solely to enhance prediction performance but, more crucially, to leverage the important and implicit WAT parameters within the sub-clusters for yield improvement purposes.

These studies collectively contribute to understanding how challenges in managing the SECOM dataset for semiconductor manufacturing processes have been addressed, providing insights into feature selection, class imbalance, and classification algorithm selection.

3. Methodology and Implementation

3.1. Data Preprocessing

The UCI SECOM dataset is derived from a semiconductor manufacturing process encompassing over one hundred steps, and it serves as a representative model for a broader category of contemporary sensor-based manufacturing processes. The dataset selected in this case is obtained from a test conducted on diverse samples of silicon wafers, aiming to identify defective ones. A total of 1567 silicon wafer examples from semiconductor manufacturing processes are included in the dataset, captured at various timestamps. Each example represents a distinct production entity, accompanied by measured features derived from signals collected by sensors and process measurement points. For every data point, there are 590 features, crucial in determining the outcome of the conducted test. Additionally, the dataset is a labeled multivariate dataset, with the last attribute forming a 1567x591 matrix, serving as the label column that indicates the pass/fail status of the test. Here, a value of -1 signifies a pass (defective wafer), while a value of 1 signifies a Fail (non-defective wafer) for that specific test data point. Furthermore, the dataset exhibits class imbalance, with only 104 instances of non-defective wafers and the remaining 1463 instances representing defective wafers. As is typical in real-life data scenarios, this dataset contains null values, their prevalence varying depending on the features of each individual instance. Addressing these null values becomes imperative during the data investigation in the pre-processing steps.

The initial step for more intricate analyses and effective model building is to comprehend the distribution of data points and analyze the inherent structures and patterns within the data. One commonly employed algorithm for this purpose is K-means clustering, which aids in assessing the complexity of a dataset. The K-means clustering result is visualized by plotting the data points and their assigned clusters. The optimal number of clusters is determined using the Elbow Method, and the K-means clustering result for the optimal K is depicted in Fig. 1.

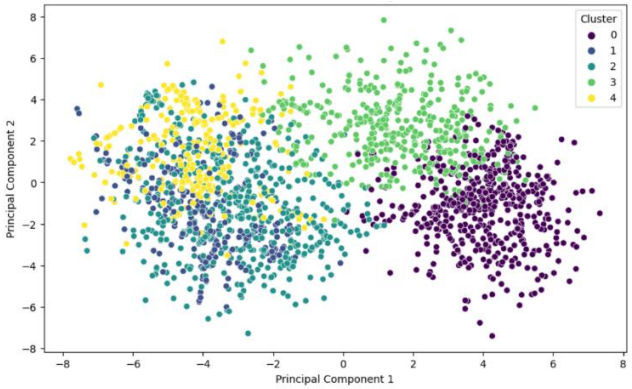

Fig. 1. K Means Clustering result for optimal K value.

Since this paper focuses on the binary classification of data, the overlapping of clusters implies a subtle boundary of difference between the classes. Furthermore, outliers exist and hold significance; therefore, they should be appropriately addressed in the models. Consequently, the dataset is intricate, and its features exhibit close interrelation, underscoring the need for an appropriate data preprocessing sequence to achieve optimal results. Therefore, the dataset poses challenges like missing data, noise, a large number of features, and class imbalance. Fig. 2 outlines the proposed steps to address these issues in the SECOM dataset.

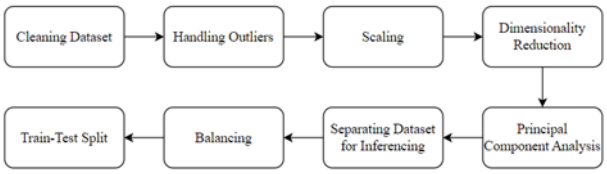

Fig. 2. Implemented sequence of Data-preprocessing steps

The dataset cleaning process addresses missing values, duplicates, and inconsistencies, ensuring the dataset's quality and reliability for effective analysis and modeling. The dataset comprises a substantial number of missing or null values (NaN) which may result in information loss and holds the potential to introduce bias in parameter estimation. Specifically, approximately 4.57% of entries are marked as undefined values in the dataset.

Given the dataset's small size, the removal of rows to address missing values is impractical, as all rows are essential for making predictions. Moreover, employing imputation or interpolation to handle missing values may yield inaccurate conclusions. As a pragmatic solution, all missing values are uniformly replaced with a constant value

of zero. Further, the removal of constant value features has been performed because these features have zero variance, which means there is no variability in that feature across the dataset. Such features do not contribute to the model's ability to distinguish between different instances. By removing constant value features from the dataset, the features were reduced from 590 to 478.

During the dataset evaluation, the presence of outliers was observed, potentially distorting the learning process of machine learning models and impacting performance by introducing noise and bias. To effectively address outliers, box plots offered a comprehensive visualization of data distribution, facilitating clear identification of potential outliers. The box plot showcasing outliers is presented in Fig. 3.

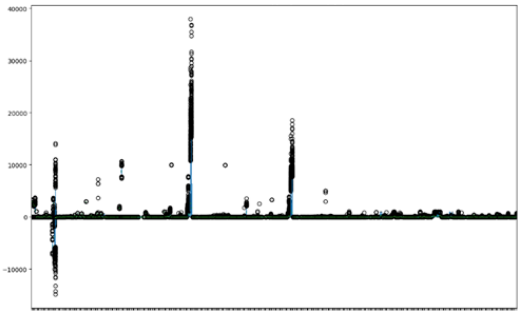

Fig. 3. Box plot of dataset features before handling outliers.

Handling outliers is crucial for models to generalize well to new, unseen data. Methods like trimming and capping can be employed, and in this approach, the capping method is preferred due to the dataset's small size. Removing outliers might further diminish data points, hindering accurate model building.

The capping method, applied to mitigate the impact of extreme values and skewed data distribution, involves setting the interquartile range (IQR) for each feature. It identifies outliers beyond defined bounds and adjusts them by capping values below the lower bound (LB) or above the upper bound (UB) to their respective bounds. Fig. 4 displays the box plot depicting the dataset after handling outliers using the capping method.

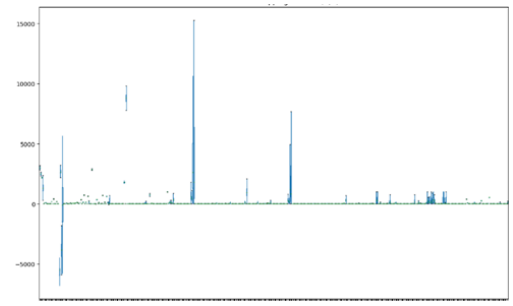

Fig. 4. Box plot of dataset features after handling outliers by capping method.

Scaling is essential for the application of various techniques, ensuring equitable contributions from features with diverse scales during model training. This process enhances the model's performance by preventing certain features from dominating others during the learning process. Standard scaling has been employed for normalizing data, establishing a mean of 0 and a standard deviation of 1 across features, thereby ensuring uniform scaling and contributing to enhanced model robustness.

Feature reduction is crucial to prevent overfitting, especially when the number of features are large (590 features) and fewer samples are present in the dataset. Overfitting arises when a model learns the noise in the data instead of the underlying patterns, giving rise to the curse of dimensionality. By emphasizing the most relevant features, feature reduction serves as a strategy to alleviate the risk of overfitting.

One method of dimensionality reduction adopted is by removal of highly correlated features. This step eliminates features that exhibit strong correlations that might redundantly convey similar information, potentially leading to multicollinearity issues. The process entails identifying feature pairs with correlation coefficients exceeding 0.9 and retaining only one feature from each correlated pair. Through this method, the number of features was reduced from 478 to 276.

Additionally, principal component analysis (PCA) is applied to convert a set of uncorrelated features or variables into a set of correlated ones which are called principal components, using orthogonal data linear transformations. Three approaches exist for determining the number of components: selecting those with eigenvalues greater than 1, opting for factors that explain at least 80% of the variance, or continuing until a break occurs in the graph. In this experiment, the second method is utilized, selecting the first 100 principal components, capturing the maximum variance within the data. The corresponding box plot for all PCA is illustrated in Fig. 5.

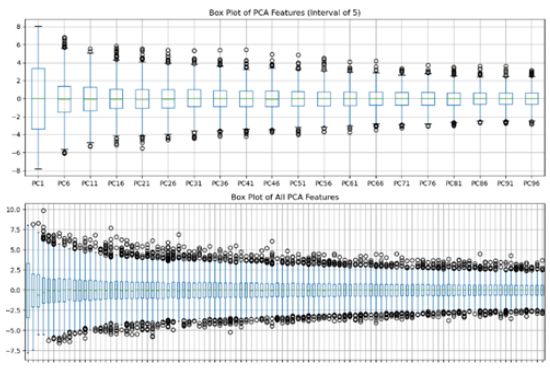

Fig. 5. Box plot of PCA features.

Before applying different models to the dataset 10 instances

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (2)$$

from both classes are removed from the dataset for inferencing. It is a strategic step often taken to enhance the performance of a model. By doing so, we aim to improve the model's ability to generalize well to new, unseen data, resulting in a more robust and accurate predictive tool.

The dataset exhibits class imbalance, with a ratio of approximately 1:14, having fewer instances of non-defected wafers. This imbalance contributes to overfitting, where a model memorizes majority class instances but struggles to generalize to new instances, particularly those from the minority class. To address this issue in a small-sized dataset, a widely adopted approach involves oversampling the Minority Class, ensuring 1452 samples for both classes.

This is accomplished using the Synthetic Minority Oversampling Technique (SMOTE), which generates synthetic instances for the minority class by interpolating between existing minority class instances, effectively mitigating class imbalance problems. Subsequently, the dataset is partitioned into training and testing sets at a ratio of 7:3. This ensures that the model is trained on a substantial portion of the data, setting the stage for subsequent model building and evaluation.

3.2. Development of Fault Detection ML and DL Models

Numerous Machine Learning and Deep Learning models within the AI field have been developed to analyze the provided training dataset. Following development, these models undergo rigorous testing to evaluate their accuracy. Subsequently, specific ensemble methods tailored to each model are employed to optimize their performance. This section provides a comprehensive elaboration on the various models developed, including the hyperparameters selected for the dataset, with the aim of optimizing the performance of each model.

3.2.1. K-Nearest Neighbours

Commonly employed for classification tasks in machine learning. It belongs to the category of instance-based learning algorithms, K-Nearest Neighbors (KNN) is a widely used supervised learning algorithm, where the model essentially memorizes the training dataset D. When a new data point Z = (x', y') is introduced, the algorithm computes the distances between x ′ and all other data points x using a suitable distance metric, and selects the k-nearest neighbors set Dz for point Z. The target class label/output for point Z is represented as in (1) for simplification. here (xj, yj) ϵ Dz.

$$y' = mode(y_i) \tag{1}$$

KNN operates as a lazy learner algorithm, which may pose computational challenges for large size datasets. The default distance metric used in scikit-learn's KNN model is Euclidean distance because of its computational efficiency and ease of implementation. The performance of KNN is highly influenced by the choice of the hyperparameter 'k', which directly impacts the model's performance and generalization ability. Therefore, the optimal value of 'k' is determined by identifying the value that minimizes the error rate across a range of values from 1 to 50. This iterative process ensures that the KNN model achieves a balanced trade-off between bias and variance, enhancing its overall performance.

3.2.2. Logistic Regression

A supervised algorithm using a probabilistic approach for binary classification. It models the relationship between independent variables and the probability of a binary outcome using the logistic function. Here (2) it uses the sigmoid function to map real-valued number into the range [0, 1].

Where, x is the linear combination of input features and their respective weights.

Model parameters are learned from training data using optimization techniques like maximum likelihood estimation or gradient descent. The Logistic Regression model built for the preprocessed dataset is trained with a 500-iteration limit with an objective to optimize accuracy. The performance is enhanced by applying boosting ensemble methods to models using AdaBoostClassifier to improve the performance of weak learners and create a strong predictive model. It is particularly effective for binary classification. The integration enhanced predictive capabilities by mitigating overfitting, reinforcing accuracy and contributing to overall reliability in die-level failure prediction.

3.2.3. Naïve Bayes

Rooted in Bayes theorem, which calculates the probability of a class given the observed features. Naive Bayes assumes feature independence, which may not hold true in real-world data but simplifies calculations and performs well in many practical scenarios. The algorithm computes the probabilities of each feature occurring given a class and the prior probability of each class, as shown in (3). The predicted class for a given input is determined by selecting the class with the highest probability. Further,

$$P\left(\frac{A}{B}\right) = \frac{P(B|A) * P(A)}{P(B)} \tag{3}$$

AdaBoostingClassifier boosting method is used along with naive bayes to enhance the predictive capabilities of Naive Bayes similar to logistic regression.

Where, P(A|B) represents the posterior probability, P(A) denotes the prior probability, P(B|A) signifies the likelihood, P(B) stands for evidence.

3.2.4. Decision Tree

This Classifier was chosen for its ability to uncover nuanced relationships, its ability to handle complex patterns, and offers clear visualizations. The default Gini impurity

$$Gini = 1 - \sum_{i=1}^{n}(p_i)^2 \quad (4)$$

criterion, used by the Decision Tree Classifier, is calculated as shown in (4).

Where, 'p_i' is the probability of an object being classified to a particular class. A lower Gini Index signifies a purer node, leading to clearer class distinctions. The maximum depth, a hyperparameter set to 10, controls the tree's levels, reducing complexity and preventing overfitting. This choice aids in generalization to unseen data. The transparency of decision trees facilitates an understanding of the learned decision boundaries.

To improve the accuracy of the Decision Tree further, Gradient Boosting was implemented. It is an ensemble method that sequentially constructs trees, each rectifying its predecessor's errors. Each boosting iteration fits a new tree to the residuals of the current model's predictions. The residual for each instance is the gradient of the loss function

$$Residual_i = -\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)} \quad (5)$$

with respect to the predicted value as represented in (5).

Where, L is the loss function. y is the true label, for instance, F(x)is the current prediction for instance i. The model's predictions are updated by a fraction of the new tree's predictions. The update formula for the m-th boosting iteration is described in (6).

$$F_m(x) = F_{m-1}(x) + \eta . T_m(x) \quad (6)$$

Where, F(m) is the prediction of the ensemble at m-th iteration, T_m is the prediction of the new tree.

This process is repeated for a predefined number of iterations which is 200 for this model. The final prediction is the sum of all trees' predictions. Despite the mathematical complexities, popular libraries abstract these details, making Gradient Boosting user-friendly. However, it's important to note that this process was time-consuming, which might impact the efficiency of real-time applications. Nevertheless, the trade-off between time and accuracy is often considered worthwhile in many data-driven domains.

3.2.5. Random Forest

The Random Forest Classifier, an ensemble of decision trees, was selected for its robust predictive performance. During training, the algorithm processes a dataset containing both pass and fail dies, constructing an ensemble of decision trees through bootstrapped sampling and variable randomness. The ensemble's strength lies in its ability to yield accurate and reliable predictions through the collective insights of individual trees. For predicting the failure likelihood of a new die, the algorithm computes a score by averaging the sub-scores across all trees. Each tree's sub-score is based on the fraction of failed instances within its corresponding leaf node.

Random Forests are typically less susceptible to hyperparameter tuning compared to individual decision trees. In this implementation, we opted for a classifier with 200 decision trees, which yielded high accuracy. This model was capable of capturing complex relationships and demonstrated resilience to noise and outliers.

3.2.6. Support Vector Machine

The employment of a SVM with a radial basis function (RBF) kernel is motivated by its robust capability to manage

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i exp(-\gamma||x - x_i||^2) + b \quad (8)$$

$$K(x, x_i) = exp(-\gamma||x - x_i||^2) \quad (7)$$

intricate, non-linear relationships within data. The RBF kernel is defined as in (7).

Where, x and x_i are data points, and γ is a parameter controlling the shape of the decision boundary. This introduces flexibility by implicitly mapping data into a higher-dimensional space, thereby enabling the SVM to discern complex patterns and create optimal decision boundaries, which are crucial for nuanced classification tasks. The SVM formulation seeks to find the hyperplane with maximum margin, expressed as (8).

Here, α_i are the dual coefficients obtained during training, y_i are the class labels and b is the bias term.

The inherent regularization parameter in SVMs along with RBF kernel aids as a potent tool for achieving accurate and reliable classification, particularly when confronted with challenging and non-linear datasets.

3.2.7. Fully connected Neural Network

An ANN is utilized due to its adeptness in handling complex relationships in binary classification tasks. The model is designed to mitigate overfitting, particularly for small datasets. The ANN model comprises three hidden layers with 512, 128, and 128 neurons, respectively as shown in Fig. 6.

Each hidden layer is succeeded by a Rectified Linear Unit

$$f(x) = \max(0, x) \quad (9)$$

(ReLU) activation function, defined as (9),

This introduces non-linearity to the network. The HeUniform initializer is used for weight initialization, enhancing training stability. Each hidden layer incorporates batch normalization and dropout regularization (with a dropout rate of 0.5) to improve generalization, speed up the training and reduce overfitting. The output layer features a single neuron with a sigmoid activation function, suitable for binary classification as shown in (2). The model is trained using the Adam optimizer, minimizing the binary cross-entropy loss as in (10).

Where, y represents true labels and $\hat{y}$ denotes predicted

$$L(y, \hat{y}) = -(y.\log(\hat{y}) + (1-y).\log(1-\hat{y})) \quad (10)$$

probabilities.

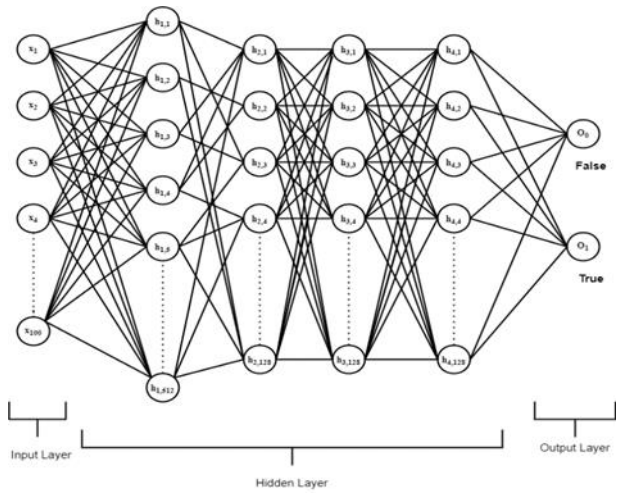

Fig. 6. Deep Neural Network.

The choice of a limited number of layers aligns with the characteristics of the small dataset. The network's ability to learn and generalize on the given task is evaluated over 20 epochs, based on training and testing accuracies. These elements incorporated in the neural network design ensure effective learning and robust performance in the context of a limited dataset.

4. Result

A comprehensive analysis was conducted to identify the most effective algorithm for the preprocessed dataset. The results from analyzing classification report, confusion matrix, The Receiver Operating Characteristic-Area Under the Curve (ROC-AUC) curve, test and train accuracies for all models are summarized in Table 1.

Table 1. Performance metrics of AI models

Model	Precision	Recall	F1-Score	Train Accuracy	Test Accuracy
KNN	1.000	0.580	0.730	1.000	0.788
Logistic Regression	0.880	0.750	0.810	0.873	0.823
LR with Boosting	0.790	0.770	0.780	0.835	0.780
Naïve Bayes	0.750	0.780	0.770	0.786	0.759
NB with Boosting	0.790	0.790	0.790	0.814	0.788
Decision Tree	0.790	0.790	0.790	0.965	0.842
DT with Boosting	0.980	0.940	0.960	1.000	0.963
Random Forest	0.990	1.000	0.990	1.000	0.991
SVM	0.990	0.990	0.990	1.000	0.994
Neural network	1.000	0.970	0.990	0.993	0.986

The results reveal that the SVM outperforms all other models, achieving a test accuracy of 99.4%, a training accuracy of 100%, and an F1 score, recall, and precision of 0.99. The SVM's high performance can be attributed to the use of radial kernel functions. The ROC-AUC curves, as depicted in Fig. 7, which shows the ROC-AUC curves for all the AI models, confirms the SVM model's accuracy, reliability and inclines towards a perfect classifier.

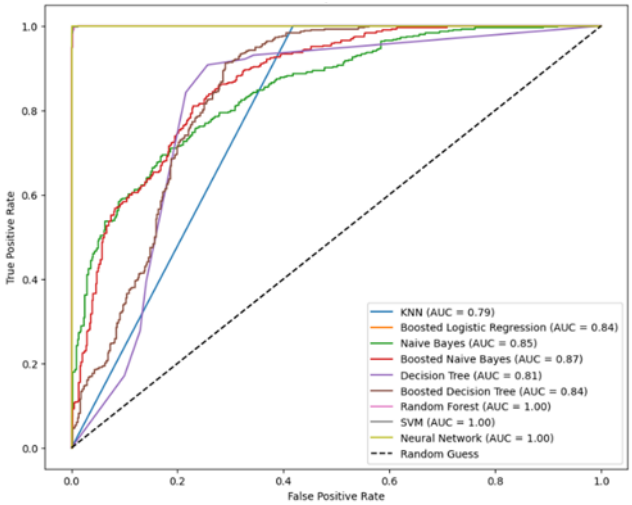

Fig. 7. ROC-AUC curves for all the AI Models.

Next in line is the Random Forest classifier, which demonstrates strong performance with a test accuracy of 99.1% and a training accuracy of 100%. The close match between training and test accuracies suggests minimal overfitting, underscoring the reliability and effectiveness of the Random Forest classifier. Despite requiring more

training time than other algorithms, the Random Forest, as an ensemble technique, enhances accuracy through iterative learning.

The Neural Network, achieving a test accuracy of 98.6%, a training accuracy of 99.3%, and a precision of 100%, indicates no False Positive Predicted cases. The architecture, employing dense layers with ReLU activation, batch normalization, and dropout, captures complex data patterns effectively. The Decision Tree exhibits a test accuracy of about 84.17% and a training accuracy of 96.45%. Gradient boosting enhances the Decision Tree's performance, achieving a test accuracy of 96.3% and training accuracy of 100%.

Other models, such as Logistic Regression and Naive Bayes, yield training and test accuracies ranging from 79% to 84%. Both Naive Bayes and Logistic Regression assume independence among features, and their linear nature might struggle to discern intricate non-linear relationships within the data.

These assumptions can limit their effectiveness in capturing complex, non-linear relationships and dependencies present in the dataset.

The KNN model, optimal for fewer features and linear data, shows a training accuracy of 100% but a test accuracy of 78%, suggesting overfitting due to the high feature count and non-linear separability of the data. Despite its accuracy, KNN is slower and more memory-intensive, requiring storage of the entire training dataset for prediction. Additionally, its reliance on Euclidean distance is very sensitive to magnitudes, hence features in the dataset that has high magnitudes will always weigh more than their counterparts with low magnitudes.

The data points that were initially removed for inference were individually fed into the model for evaluation. The results, as shown in Table 2, provide insights into the performance of various machine learning models on these instances.

Table 2. Inference Results of AI models

Model	No. of True Predictions	No. of False Predictions			Inference Accuracy (%)
		Non-Defected	Defected	Total	
KNN	14	3	6	9	60.870
Logistic Regression	15	7	1	8	65.217
Naïve Bayes	15	5	3	8	65.217
Decision Tree	14	8	1	9	60.870
Random Forest	12	11	0	11	52.174
SVM	13	10	0	10	56.522
Neural Network	14	9	0	9	60.870

In the analysis, KNN, Logistic Regression, Naive Bayes, and Decision Tree models achieved inference accuracies of 60.87%, 65.22%, 65.22%, and 60.870% respectively. Despite these relatively high accuracies, a closer examination of the number of misclassified defected and non-defected instances suggests that these models may be predicting values randomly rather than identifying specific classes accurately. The Random Forest model, with an inference accuracy of 52.17%, predicted all values as misclassified, which corresponds to the majority class.

The SVM and Neural Network models accurately identified the defected class and a subset of the non- defected class. The oversampling process of the non-defected class introduced synthetic data with limited variation, which consequently led to an inadequate capture of this class patterns. This observation suggests that the use of a larger and more representative dataset for training could potentially enhance the efficiency of SVM and Neural Network models. From this comparative analysis of both results and inference, it is evident that SVM and Neural Network models outperform all other models.

5. Conclusion

In the critical task of identifying and classifying defective wafers in semiconductor manufacturing, this paper suggests a preprocessing approach that includes dimensional reduction, outlier handling, and balancing. These steps significantly enhance the efficiency of various Machine Learning and Deep Learning models, with the ultimate goal of identifying the most effective algorithm for the dataset. After evaluating various models, Neural Networks and SVM emerge as the most promising due to their high performance and minimal overfitting. While SVMs offer high accuracies, Neural Networks show potential for exceptional performance, especially with large datasets and real-time applications. Although Random Forest achieves high accuracy, its computational demands make it impractical for real-world applications.

For more practical applications, we suggest that Neural Networks and SVMs are better. They strike a balance between accuracy, efficiency, and reliability, making them optimal for practical deployment in wafer classification. Looking ahead, this research can be expanded to real-time monitoring and inspection using machine vision systems. These systems can detect processing errors and enable rapid correction of the processing parameters in real-time with minimum human intervention. Additional algorithms can be

developed for classifying various types of defects. This will provide valuable insights for optimizing processes by identifying and categorizing defect types and their frequencies occurring in silicon wafers. This approach promises to significantly improve the efficiency and effectiveness of semiconductor manufacturing.

References

[1] S. Munirathinam and B. Ramadoss, "Predictive models for equipment fault detection in the semiconductor manufacturing process," *IACSIT International Journal of Engineering and Technology*, vol. 8, no. 4, pp. 273–285, 2016.

[2] A. Kaveri Sharma, "A study on effects of intrinsic characteristics of datasets on classification performance," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 6, no. 1, pp. 198–204, 2016.

[3] J.Wang, Z. Yang, J. Zhang, Q. Zhang, and W.-T.-K. Chien, "AdaBalGAN: An improved generative adversarial network with imbalanced learning for wafer defective pattern recognition," *IEEE Trans. Semiconductor Manuf.*, vol. 32, no. 3, pp. 310-319, Aug. 2019.

[4] M. Saqlain, B. Jargalsaikhan, and J. Y. Lee, ''A voting ensemble classifier for wafer map defect patterns identification in semiconductor manufacturing,'' *IEEE Trans. Semicond. Manuf.*, vol. 32, no. 2, pp. 171–182, May 2019.

[5] K.-J. Kim, K.-J. Kim, C.-H. Jun, I.-G. Chong, and G.-Y. Song, ``Variable selection under missing values and unlabeled data in semiconductor processes," *IEEE Trans. Semiconductor Manuf.*, vol. 32, no. 1, pp. 121-128, Feb. 2019.

[6] J.-S. Kim, S.-J. Jang, T.-W. Kim, H.-J. Lee, and J.-B. Lee, "A productivity-oriented wafer map optimization using yield model based on machine learning," *IEEE Trans. Semiconductor Manuf.,* vol. 32, no. 1, pp. 39-47, Feb. 2019.

[7] J. Wang, J. Zhang, and X. Wang, ''A data driven cycle time prediction with feature selection in a semiconductor wafer fabrication system,'' *IEEE Trans. Semicond. Manuf.*, vol. 31, no. 1, pp. 173–182, Feb. 2018.

[8] K. Nakata, R. Orihara, Y. Mizuoka, and K. Takagi, "A comprehensive Big-Data-Based monitoring system for yield enhancement in semiconductor manufacturing," *IEEE Trans. Semiconductor Manuf.*, vol. 30, no. 4, pp. 339-344, Nov. 2017.

[9] S. Kang, S. Cho, D. An, and J. Rim, ''Using wafer map features to better predict die-level failures in final test,'' *IEEE Trans. Semicond. Manuf*, vol. 28, no. 3, pp. 431–437, Aug. 2015.

[10] Y. Meidan, B. Lerner, G. Rabinowitz, and M. Hassoun, ''Cycle-time key factor identification and prediction in semiconductor manufacturing using machine learning and data mining,'' *IEEE Trans. Semicond. Manuf.*, vol. 24, no. 2, pp. 237–248, May 2011.

[11] S.-J. Jang, J.-H. Lee, T.-W. Kim, J.-S. Kim, H.-J. Lee, and J.-B. Lee, ``A wafer map yield model based on deep learning for wafer productivity enhancement," *in Proc. 29th Annu. SEMI Adv. Semiconductor Manuf. Conf. (ASMC)*, Apr. 2018, pp. 29-34.

[12] Y. Kong and D. Ni, "A practical yield prediction approach using inline defect metrology data for system-on-chip integrated circuits,*" in Proc. 13th IEEE Conf. Autom. Sci. Eng. (CASE),* Aug. 2017, pp. 744-749.

[13] D. Moldovan, T. Cioara, I. Anghel and I. Salomie, "Machine learning for sensor-based manufacturing processes," *2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, Cluj-Napoca, Romania, 2017, pp. 147-154, doi: 10.1109/ICCP.2017.8116997.

[14] S. H. Park, C.-S. Park, J. S. Kim, S.-S. Kim, J.-G. Baek, and D. An,''Data mining pproaches for packaging yield prediction in the postfabrication process,*'' in Proc. IEEE Int. Congr. Big Data,* Jun. 2013, pp. 363–368.

[15] K. Kerdprasop and N. Kerdprasop, "Feature selection and boosting techniques to improve fault detection accuracy in the semiconductor manufacturing process," *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 1, 2011.

[16] D. Jiang, W. Lin and N. Raghavan, "A Gaussian Mixture Model Clustering Ensemble Regressor for Semiconductor Manufacturing Final Test Yield Prediction," *in IEEE Access*, vol. 9, pp. 22253-22263, 2021, doi: 10.1109/ACCESS.2021.3055433.

[17] D. Jiang, W. Lin and N. Raghavan, "A Novel Framework for Semiconductor Manufacturing Final Test Yield Classification Using Machine Learning Techniques," *in IEEE Access*, vol. 8, pp. 197885-197895, 2020, doi: 10.1109/ACCESS.2020.3034680.

[18] T. Ishida, I. Nitta, D. Fukuda, and Y. Kanazawa, "Deep learning-based wafer-map failure pattern recognition framework,*" in Proc. 20th Int. Symp. Qual. Electron. Design (ISQED),* Mar. 2019, pp. 291-297.

[19] J. Kim, Y. Han, and 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.

[20] H. F. Jelinek, A. Yatsko, A. Stranieri, S. Venkatraman, and A. Bagirov, "Diagnostic with incomplete nominal/discrete data," *Artificial Intelligence Research*, vol. 4, no. 1, pp. 22–35, 2015.

[21] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[22] T. Chen, ''A PCA-FBPN approach for job cycle time estimation in a wafer fabrication factory,'' *Int. J. Fuzzy Syst. Appl.*, vol. 2, no. 2, pp. 50–67, Apr. 2012.

[23] J. Mendes-Moreira, C. Soares, A. M. Jorge, and J. F. D. Sousa, ''Ensemble approaches for regression: A survey,'' *ACM Comput. Surv.*, vol. 45, no. 1, pp. 1–40, Nov. 2012.

[24] S. Mahadevan and S. L. Shah, ''Fault detection and diagnosis in process data using one-class support vector machines,'' *J. Process Control*, vol. 19, no. 10, pp. 1627–1639, Dec. 2009.