# Semiconductor Bearing Fault Recognition

**Nikhita Mishra, Ipshitta Chaturvedi, Janhvi Mehta**

***Abstract:** Semiconductor manufacturing is consid-ered to be one of the most technologically complicated manufacturing processes. Bearing, being a critical part of the rotating machinery used in the process, plays an essential role as it supports the mechanical rotating body and decreases the friction coefficient. However, extensive use makes this element a target of health degradation, which indirectly causes machine failure. A defective bearing causes approximately 50% of failures in electrical machines. Hence, there arises a dire need for effective fault detection and diagnosis methods to recog-nise fault patterns and help take preventive measures. This paper carries out a comprehensive comparative study of the pre-existing machine learning and deep learning techniques used for diagnosing bearing faults and further devises a novel framework for bearing fault diagnosis based on the results. Unlike the conventional Fault Detection Classifiers (FDC) that operate in the original data space, this algorithm explores the scope for feature extraction and transferability empowered by the deep learning models used.*

***Keywords:** semiconductor manufacturing, defective bearing, machine learning, deep learning.*

## I. INTRODUCTION

Semiconductors, labelled under the category of microelectronics, are employed at the crux of a majority of electronic systems and shall continue to do so in the upcoming technologies in both-the consumer as well as industrial markets. Their wide application is owed to their inexpensiveness, dependability, compactness and efficiency. The processing of semiconductor wafers, as a part of semiconductor manufacturing, undergoes several stages, namely- cleaning, film deposition, resist coating, exposure, development, etching, impurity insertion, activation, assembly and packaging. All the steps of this process have different sensors associated with them, which compute parame-ters fulfilling various purposes. Alongside these,several other parameters are generated from the workstation as well as the equipment throughout the manufacturing procedure. This vast base of attribute sources, when combined with the pre-existent complexity of the manufacturing process, gives birth to a colossal amount of data which can prove substantially beneficial in digitising and facilitating the process.

Manuscript received on August 21, 2021.
Revised Manuscript received on August 31, 2021.
Manuscript published on October 30, 2021.
* Correspondence Author

**Nikhita Mishra***, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore (Tamil Nadu), India.
**Ipshitta Chaturvedi**, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore (Tamil Nadu), India.
**Janhvi Mehta**, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore (Tamil Nadu), India.

The machinery used in the manufacturing of semiconductors may have to work under adverse conditions like high ambient temperature, high moisture, and overload, which slowly turn into motor malfunctions that lead to high maintenance costs, substantial financial losses, and safety haz-ards. These malfunctions can be caused due to faults of various categories, one of them being bearing faults.Many real-life observations show that the continuous wearing down is due to the relative motion between surfaces which causes damage. Studies have concluded that bearing is the root cause of the problem. It is imperative to combat this issue for the welfare of the machinery as well as the labour. These issues justify the need for effective fault detection and diagnosis methods to obtain the healthy conditions of bearings and recognise the fault patterns. Hence, the proposed algorithm provides a solution for fault detection of bearings at an early stage, which will further eradicate the manual intensive job of finding de-fects in bearings and help maintain the health of machinery.

In traditional methods for bearing fault detec-tion, it is common to have a multivariate data set. The measured signals contain both useful and irrelevant information. The SECOM dataset contains 1467 total instances and 590 anonymised features. The output of the process is a simple pass/fail response. On a quick glance, it can be noticed that the dataset is riddled with missing data, and only a few columns contain most of the missing data.

The conventional approaches for bearing fault detection diagnosis using vibrational signals in-clude three steps: data preprocessing, feature ex-traction, and pattern classification. The challenge for the success of these models lies in the choos-ing of features. Features like the time domain or the frequency domain have limitations that pose problems in detecting the faulty component. The machine learning models can only describe the signal features of some well-defined fault types, while practically, the naturally occurring faults are much more complex. Therefore, patterns or unique features may exist in the data, potentially reveal-ing a bearing fault. In recent years, deep learn-ing algorithms have gained traction to meet this demand due to their several advantages, mainly as they could discover intricate structures in big data. When put in comparison with the traditional machine learning algorithms, deep learning has made immense progress be it in image recognition or speech recognition, and especially in bearing fault detection.

This paper puts forward a systematic review of the several machine learning and deep learning techniques deployed for fault detection and clas-sification in the existing literature and hence high-lights the most ideally suited algorithms optimised for bearing fault detection.

The challenges faced in datasets related to such projects are that the number of features in these datasets is enormous as the sensors are under surveillance constantly. However, not all features are equally valuable in a specific monitoring sys-tem. This study proposes an optimised model ob-tained by performing oversampling and undersam-pling of data, hyperparameter tuning, feature im-portance and dimensionality reduction producing accurate and well-rounded results. Further, upon building a deep neural network, the machine learn-ing and deep learning algorithms are compared to arrive at the combination of steps that best fits the given problem statement.

## II. RELATED WORK

The research work by Moldovan et al. [5] pro-vides a fault classification model for the SECOM dataset via a pipeline of data preprocessing stages, feature selection, data classification and sampling techniques. This paper makes use of two novel algorithms, namely the MARS and the Boruta al-gorithms, for the process of feature selection. How-ever, it fails to make use of any cross-validation techniques before applying the feature selection, and instead, cross-validation is carried out post feature selection. This might result in a biased estimate of the variance. The same oversight of applying the cross-validation afterwards has been observed in [7, 9].

In the study carried out by Munirathinam et al.[6], machine learning based methods are adopted in order to build a fault detection model with the utilisation of techniques such as feature selection, chi-square statistical analysis, oversam-pling and Principal Component analysis. While resampling the data from the minority class, to carry out oversampling of data, the model gets prone to data overfitting. There is also a possibility of attaining lower prediction accuracies due to the presence of rare cases. Obstacles like these have been handled in the study by discussing and making use of techniques like subject matter expert knowledge, variable component analysis, correlation analysis and deep belief networks. Zhang et al. [2] proposed a fault diagnosis based on the deep transfer learning method, which examined and learned the features from a huge data source. The model used this data to adjust the parameters of neural networks. Some of these parameters were transferred from the source task to the target task, hence assisting the model training on a small target data amount.

The study by Jelinek et al. [16] deals with the issue of incomplete data. Classifiers such as Na¨ıve Bayes, Nearest Neighbor and Decision Tree are used to derive the data completion procedure, and the missing values are replaced by applying statistical approaches so as to eradicate any issues in classification.

## III. METHODOLOGY

Overall, the selected dataset exhibits three com-plications, i.e., the absence of values from certain records, the presence of a multitude of features-giving rise to irrelevant features, and a data class imbalance due to the disproportionate ratio of the passed cases to the failed cases.

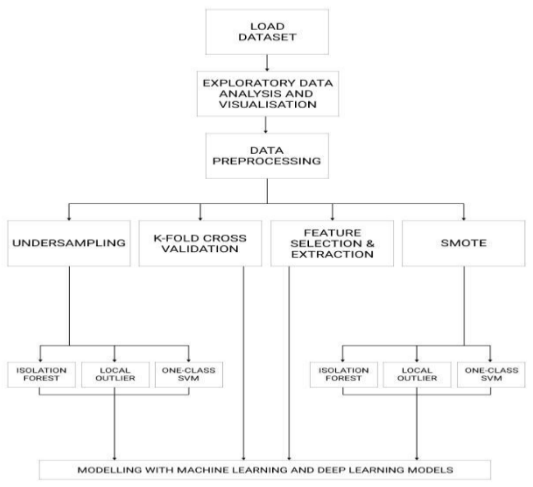


**FIG 1: Flow of the proposed algorithm.**

A. Data Preprocessing

Initially, the data is put through the data prepro-cessing phase, where data filtering, unfolding and scaling is carried out using the sklearn preprocess-ing modules. The raw dataset is taken, and data cleaning is performed by dropping the columns with constant values or a high count of missing entries and handling the columns with a lower count of missing entries. The data is then visu-alised to have a better understanding of it. Next, the dependent and independent data is separated, further splitting it into training and testing datasets. As the dataset has multiple features spanning vary-ing magnitude and ranges, feature scaling, i.e., standardisation of the data, is performed as the last step in the preprocessing of data. Other specific processes may require specific filtering techniques.

*B. Modelling*

Following the preprocessing phase, modelling of the data is done by training several machine learning models, including- Random Forest, Sup-port Vector Machine, K-Nearest Neighbors and XGBoost- to test how different models perform in the given problem. Along with the accuracy performance of each model, the anomaly detection rate is computed to, later on, compare and visualise it with the accuracies obtained on implementing the different models.

*C. Sampling*

Once the first two steps are over, undersampling and oversampling of the dataset is done, with the help of the imblearn toolbox, and it then undergoes the same modelling processes as in the previous step. The need to sample the data occurs as there was a significant class imbalance in the dataset-there are only 104 fail cases, whereas a much larger number of examples pass the test, which makes a 1:14 proportion of the failed cases to the passed cases.

1) Undersampling: The proportion of the ma-jority class (here, fail cases) gets cut down when an imbalanced dataset is undersampled to reach the amount which equates with the proportion of the minority class (pass cases). This helps in balancing the skew ratio and additionally accelerates the sensitivity (or the true positive rate) of the class in the minority.

2) Oversampling: The technique used for over-sampling the data is Synthetic Minority Over-sampling Technique (SMOTE). While in the typ-ical oversampling, the records from the minor-ity class are resampled in accordance with the majority class proportion, SMOTE instead uses the KNN algorithm to formulate synthetic data and iterates the process until the proportions of the minority and the majority classes come to an equilibrium. This helps in the broadening of the decision boundaries around the minority points, hence producing better results.

Hyperparameter tuning is also done for some models, post oversampling, to determine the right combination of parameters to maximise their per-formance.

D. K-Fold

Cross-validation helps evaluate the efficiency of machine learning models. K-folds cross-validation is one such technique that aids in attaining a less biased model by ensuring rotation of each entry between training and testing datasets. Hence, it helps in avoiding overfitting of data and reaching fair results. The SECOM dataset is divided into five folds, with one fold chosen as the testing dataset and the remaining four folds act as the training data. This process of rotationally choosing the testing data is iterated until all five folds have been considered as the testing data. Thus finally leading to each dataset observation to be viewed and used for testing.

E. Feature Selection using Dimensionality Reduc-tion

The SECOM dataset consists of 590 features, and not all of these features contribute equally towards the detection of the bearing fault and are hence less relevant. Therefore, to stream-line the process, the feature importance scores of the dataset attributes are calculated, and the features that best represent the outcome are chosen. Post that, dimensionality reduction using Principal Component Analysis (PCA) is performed to dis-card the attributes which are of less relevance to the results. This would, in turn, help attain a higher accuracy with a lower number of inputs.

F. Isolation Forest Technique

Anomaly detection in massive datasets is a diffi-cult task. This algorithm is extremely beneficial in removing outliers. The technique relies on the fact that anomalies are few in number and quite dif-ferent from the other data points. Therefore, these data points are an easy target for the algorithm. The use of the efficacious use of the isolation method makes it a high-performance model. Furthermore, this technique uses an algorithm with a low linear time complexity and a reasonable memory spec-ification. It introduces the utilisation of trees by using subsamples of fixed size. Fundamentally, the
models perform better when the dataset has no alarming outlier.

G. Local Outlier Technique

The LOF algorithm is an unsupervised method to detect outliers. It configures the local density divergent of a given data cell and compares it to its neighbours. This algorithm considers the neigh-bours having a slightly lower density as outliers. This is followed by a production of an anomaly score which represents the data cells considered as outliers.

H. One-Class SVM Technique

This method is an extension of the SVM tech-nique. It is often used in an unsupervised setting for the classification of outliers. This technique is also often called "novelty detection". The SVM model is trained on only one class which is more commonly called a "normal class". Then, it takes the property of the normal class and differentiates it from the rest of the classes and classifies differ-ent behaviours as anomalies.

I. Deep Neural Network

Finally, a Deep Neural Network (DNN) is built to detect the bearing faults by experimenting with the features. The DNN works on a trial and error basis wherein it assigns particular weights to the features to return binary outputs, and if it does not come out to be accurate, the weights are adjusted. It continues to iterate until it reaches the correct equation. The DNN here is fed with original data, undersampled data and oversampled data and the validation scores are recorded.

## IV. RESULTS & DISCUSSIONS

The SECOM dataset had a skew ratio of 14:1. Thus, just training the dataset with machine learn-ing models was not enough. Initially, the dataset was trained with K Nearest Neighbour Classifier, XGBoost, Random Forest, Support Vector Classi-fier.

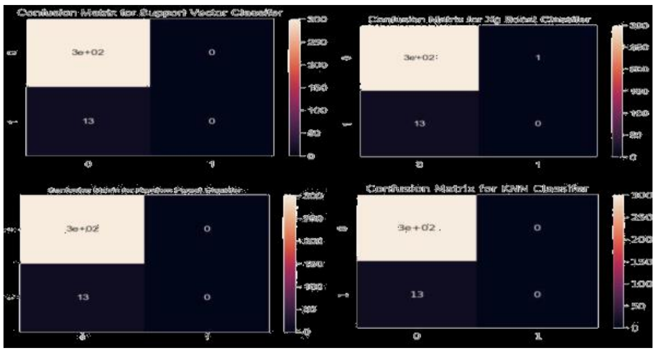


**FIG 2: Confusion matrices of (a) Support Vector Classifier (b) XGBoost Classifier (c) Random Forest Classifier and (d) KNN Classifier.**

In these confusion matrices (FIG 2), the mod-els were able to detect only 0/1 out of the 13 faults present in the dataset. Upon graphing the precision-recall curve, the PR curve comes out to be 0.4 for all the models.

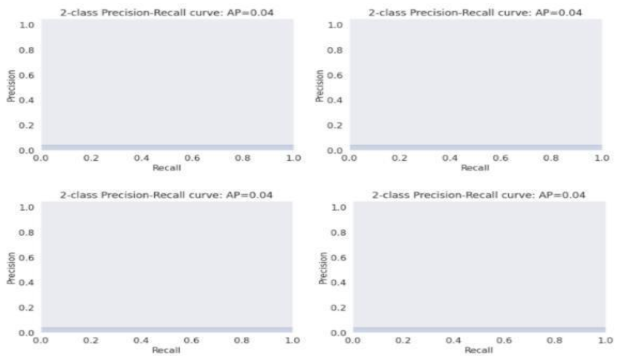


**FIG 3: Precision-Recall curves for (a) Support Vector Classifier (b) XGBoost Classifier (c) Random Forest Classifier and (d) KNN Classifier.**

While the models showed an impressive test accuracy score, the precision-recall rate was suspi-ciously low.

The undersampling and SMOTE tech-niques proved extremely fruitful as the precision-recall rate, which was previously 0.4 for all the models, shot up to 67, 68, 57, 70 per cent in Random Forest, SVC, KNN and XGB models, respectively.

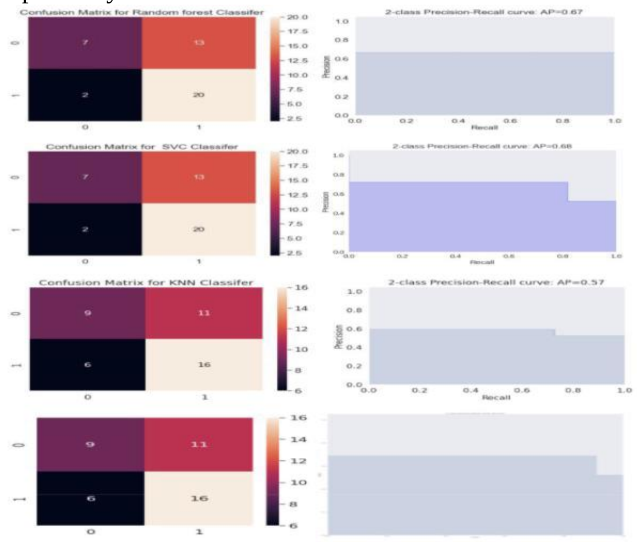


**FIG 4: Confusion matrices and Precision-Recall curves of (a) Random Forest Classifier (b) Support Vector Classifier (c) KNN Classifier and (d) XGBoost Classifier after Undersampling.**

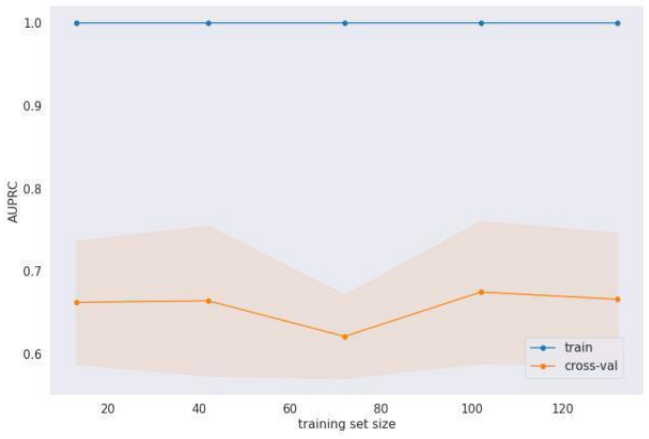


**FIG 5: AUPRC Score of the XGBoost Model.**

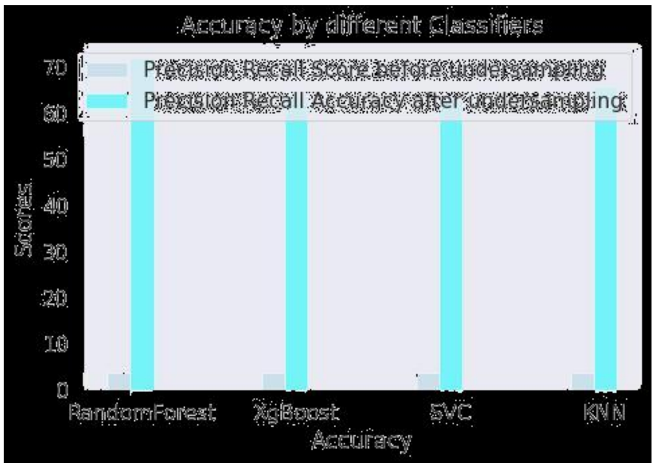


**FIG 6: Comparison of Precision-Recall rate before and after undersampling for (a) Random Forest Classifier (b) XGBoost Classifier (c) Support Vector Classifier and (d) KNN Classifier.**

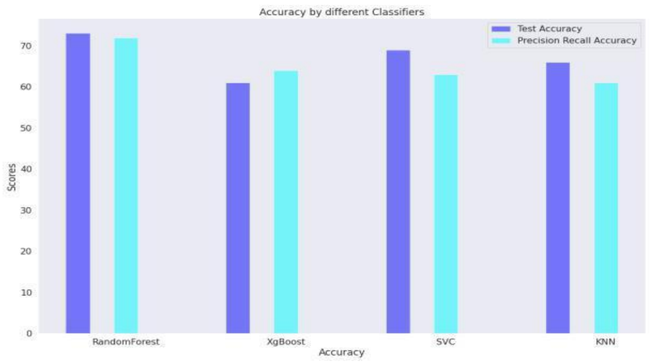


**FIG 7: Comparison of Test Accuracy versus Precision-Recall Accuracy for (a) Random Forest Classifier (b) XGBoost Classifier (c) Support Vector Classifier and (d) KNN Classifier on Undersampling.**

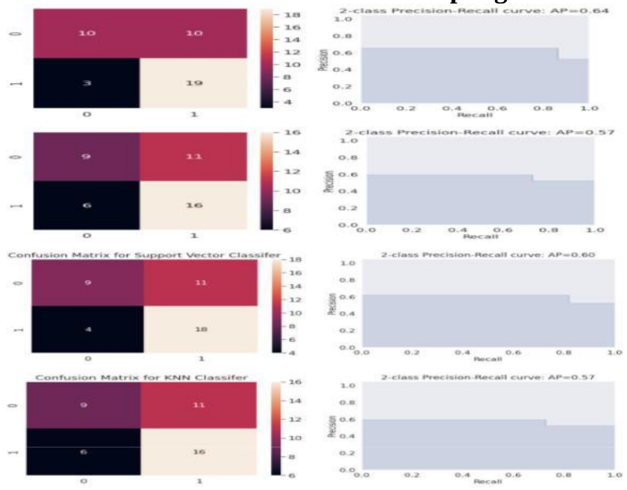


**FIG 8: Confusion matrices and Precision-Recall curves of (a) Random Forest Classifier (b) XGBoost Classifier (c) Support Vector Classifier and (d) KNN Classifier after performing SMOTE.**

After the results from the undersampling tech-nique came out to be successful, the Synthetic Minority Oversampling Technique (SMOTE) was applied to the dataset.

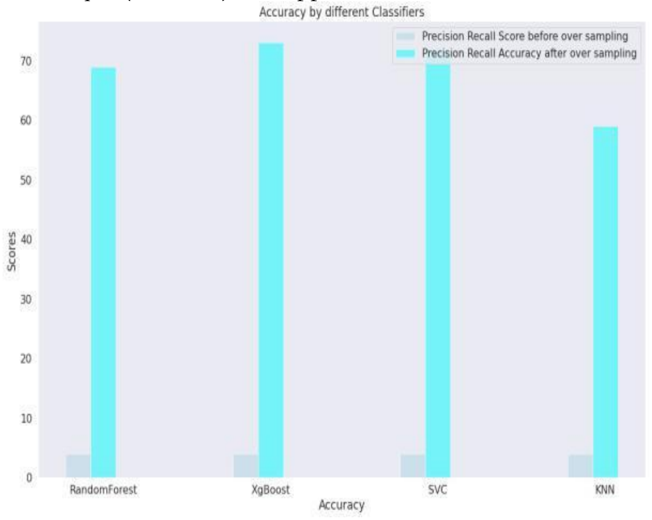


**FIG 9: Comparison of Precision-Recall rate before and after SMOTE for (a) Random Forest Classifier (b) XGBoost Classifier (c) Support Vector Classifier and (d) KNN Classifier.**

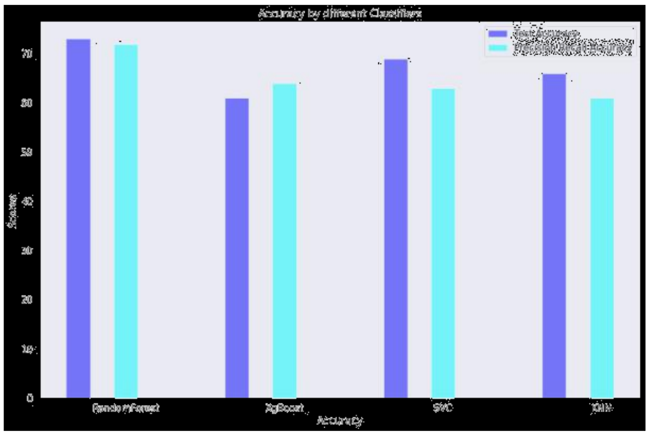


**FIG 10: Comparison of Test Accuracy versus Precision-Recall Accuracy for (a) Random Forest Classifier (b) XGBoost Classifier (c) Support Vector Classifier and (d) KNN Classifier on Oversampling.**

It is widely recognised that the K Fold Cross Validation technique gives effective results when there is a class imbalance in the dataset. Upon using the technique, the cross-validation scores were found to be worthwhile.

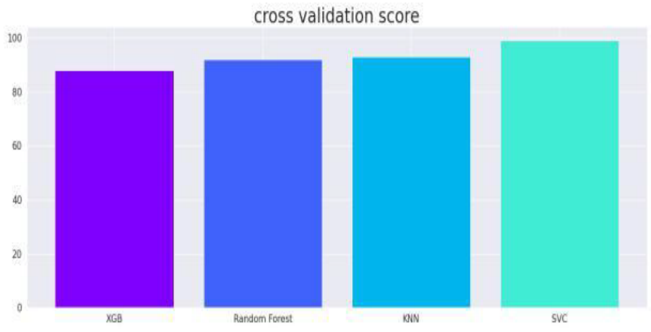


**FIG 11: Cross-validation score of (a) XGBoost Classifier (b) Random Forest Classifier (c) KNN Classifier and (d)** Support Vector Classifier after applying the K Fold Cross Validation technique. Feature importance training was then done to determine the key factors which contributed to-wards yielding excursions downstream in the pro-cess. This was followed by feature extraction via Principal Component Analysis (PCA) to convert a set of correlated features into linearly unrelated features. This helps to attain dimensionality reduc-tion, the need for which arose due to the originally high dimensionality of the dataset.

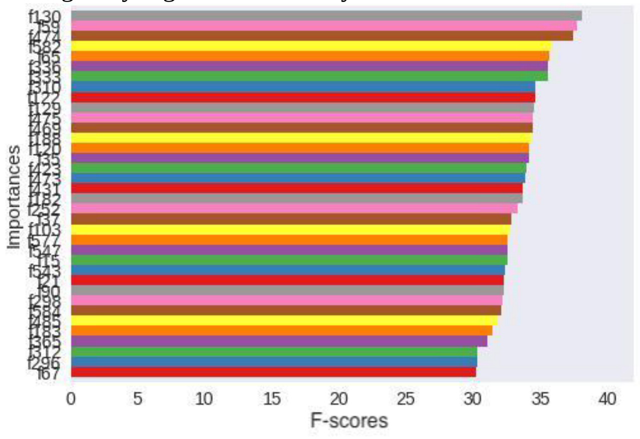
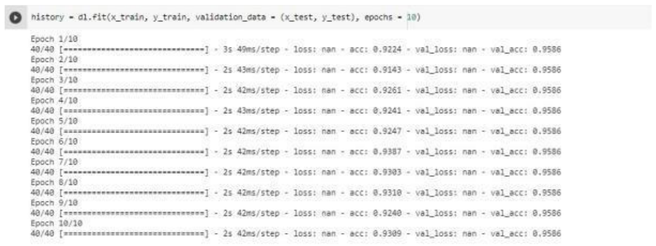


**FIG 12: The feature importance plot for the attributes of the dataset.**

Finally, a Deep Neural Network architecture was built by experimenting with different neurons and layers. First, the original dataset was put to use while building the neural network. This technique showed disappointing results as the testing score was identical for all the epochs, and there was a high validation loss score. The machine learning models exhibited this same issue as well.

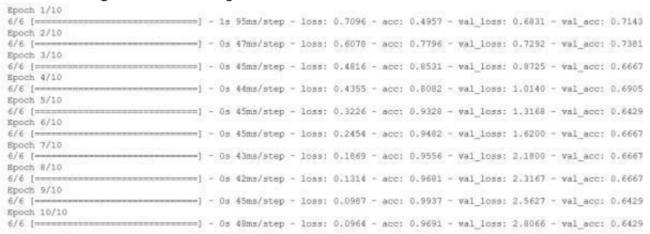


```
history = d1.fit(x_train, y_train, validation_data = (x_test, y_test), epochs = 10)

Epoch 1/10
40/40 [==============================] - 3s 49ms/step - loss: nan - acc: 0.9224 - val_loss: nan - val_acc: 0.9586
Epoch 2/10
40/40 [==============================] - 2s 43ms/step - loss: nan - acc: 0.9143 - val_loss: nan - val_acc: 0.9586
Epoch 3/10
40/40 [==============================] - 2s 42ms/step - loss: nan - acc: 0.9261 - val_loss: nan - val_acc: 0.9586
Epoch 4/10
40/40 [==============================] - 2s 43ms/step - loss: nan - acc: 0.9241 - val_loss: nan - val_acc: 0.9586
Epoch 5/10
40/40 [==============================] - 2s 42ms/step - loss: nan - acc: 0.9247 - val_loss: nan - val_acc: 0.9586
Epoch 6/10
40/40 [==============================] - 2s 42ms/step - loss: nan - acc: 0.9387 - val_loss: nan - val_acc: 0.9586
Epoch 7/10
40/40 [==============================] - 2s 42ms/step - loss: nan - acc: 0.9303 - val_loss: nan - val_acc: 0.9586
Epoch 8/10
40/40 [==============================] - 2s 42ms/step - loss: nan - acc: 0.9310 - val_loss: nan - val_acc: 0.9586
Epoch 9/10
40/40 [==============================] - 2s 42ms/step - loss: nan - acc: 0.9240 - val_loss: nan - val_acc: 0.9586
Epoch 10/10
40/40 [==============================] - 2s 42ms/step - loss: nan - acc: 0.9309 - val_loss: nan - val_acc: 0.9586
```

**FIG 13: Validation Accuracies upon feeding the original dataset to the Deep Neural Network (DNN).**

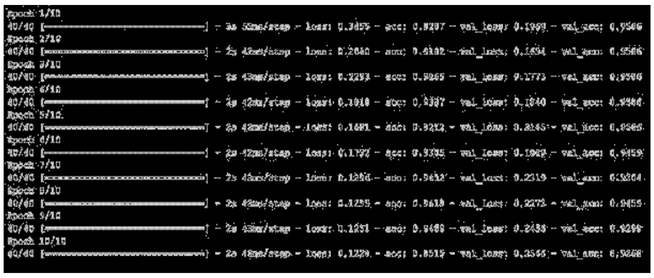
To tackle the problem of the class imbalance, an attempt at undersampling the data and applying the SMOTE technique was made again. The DNN is fed the undersampled technique first.



```
Epoch 1/10
6/6 [==============================] - 1s 95ms/step - loss: 0.7096 - acc: 0.4957 - val_loss: 0.6831 - val_acc: 0.7143
Epoch 2/10
6/6 [==============================] - 0s 47ms/step - loss: 0.6078 - acc: 0.7796 - val_loss: 0.7292 - val_acc: 0.7381
Epoch 3/10
6/6 [==============================] - 0s 45ms/step - loss: 0.4816 - acc: 0.8531 - val_loss: 0.8725 - val_acc: 0.6667
Epoch 4/10
6/6 [==============================] - 0s 44ms/step - loss: 0.4355 - acc: 0.8082 - val_loss: 1.0140 - val_acc: 0.6905
Epoch 5/10
6/6 [==============================] - 0s 45ms/step - loss: 0.3226 - acc: 0.9328 - val_loss: 1.3168 - val_acc: 0.6429
Epoch 6/10
6/6 [==============================] - 0s 45ms/step - loss: 0.2454 - acc: 0.9482 - val_loss: 1.6200 - val_acc: 0.6667
Epoch 7/10
6/6 [==============================] - 0s 43ms/step - loss: 0.1869 - acc: 0.9556 - val_loss: 2.1800 - val_acc: 0.6667
Epoch 8/10
6/6 [==============================] - 0s 42ms/step - loss: 0.1314 - acc: 0.9681 - val_loss: 2.3167 - val_acc: 0.6667
Epoch 9/10
6/6 [==============================] - 0s 45ms/step - loss: 0.0987 - acc: 0.9937 - val_loss: 2.5627 - val_acc: 0.6429
Epoch 10/10
6/6 [==============================] - 0s 48ms/step - loss: 0.0964 - acc: 0.9691 - val_loss: 2.8066 - val_acc: 0.6429
```

**FIG 14: Validation Accuracies obtained from the DNN on Undersampling.**

Finally, the SMOTE technique was used, and a validation accuracy of 92.68 was reached at epoch Additionally, validation loss scores were also observed to have been lowered.



```
Epoch 1/10
40/40 [==============================] - 3s 52ms/step - loss: 0.3655 - acc: 0.8297 - val_loss: 0.1999 - val_acc: 0.9566
Epoch 2/10
40/40 [==============================] - 2s 42ms/step - loss: 0.2040 - acc: 0.9182 - val_loss: 0.1694 - val_acc: 0.9566
Epoch 3/10
40/40 [==============================] - 2s 43ms/step - loss: 0.2293 - acc: 0.9265 - val_loss: 0.1773 - val_acc: 0.9566
Epoch 4/10
40/40 [==============================] - 2s 42ms/step - loss: 0.1918 - acc: 0.9387 - val_loss: 0.1840 - val_acc: 0.9566
Epoch 5/10
40/40 [==============================] - 2s 42ms/step - loss: 0.1681 - acc: 0.9212 - val_loss: 0.2145 - val_acc: 0.9565
Epoch 6/10
40/40 [==============================] - 2s 42ms/step - loss: 0.1792 - acc: 0.9305 - val_loss: 0.1987 - val_acc: 0.9459
Epoch 7/10
40/40 [==============================] - 2s 42ms/step - loss: 0.1256 - acc: 0.9612 - val_loss: 0.2319 - val_acc: 0.9264
Epoch 8/10
40/40 [==============================] - 2s 43ms/step - loss: 0.1255 - acc: 0.9610 - val_loss: 0.2272 - val_acc: 0.9459
Epoch 9/10
40/40 [==============================] - 2s 43ms/step - loss: 0.1251 - acc: 0.9469 - val_loss: 0.2438 - val_acc: 0.9298
Epoch 10/10
40/40 [==============================] - 2s 48ms/step - loss: 0.1220 - acc: 0.8519 - val_loss: 0.2566 - val_acc: 0.9268
```

**FIG 16: Validation Accuracies obtained from the DNN on applying SMOTE.**

After reaching such productive results, the focus shifted onto removing the anomalies in the dataset. Hence, techniques like the Isolation Forest, Local Outlier and One-Class SVM were applied.

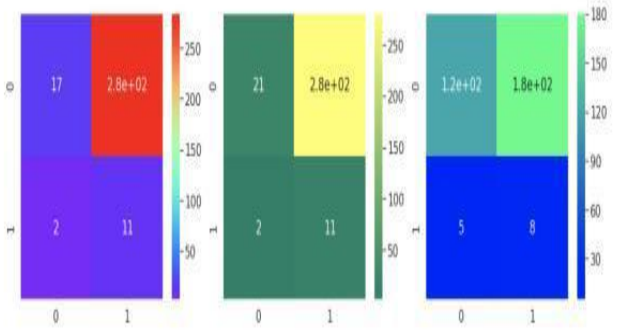


**FIG 16: (a) Isolation Forest Confusion Matrix. This model was able to detect 11 out of 13 faults. (b)**

Local Outlier Technique Confusion Matrix. This model was able to detect 11 out of 13 faults. (c) One-Class SVM Confusion Matrix. This model detected 8 out of 13 faults.

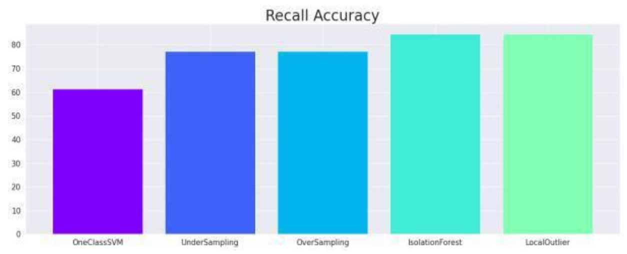


**FIG 17: Comparison of the Recall Accuracies obtained from (a) One-Class SVM (b) Undersampling (c) Oversampling (d) Isolation Forest (e) Local Outlier.**

## V. CONCLUSION

This paper examined data from a real-time semiconductor manufacturing process and diag-nosed faulty bearings. Several approaches were implemented to deal with the shortcomings of the dataset, which included class imbalance, the presence of irrelevant features and missing values. The results from these methods were analysed and compared. After sampling, It is found that XGBoost (after hyperparameter tuning) and SVC perform the best among the ML models. After the oversampled dataset underwent an artificial neural network, the validation accuracy came out to be 92.6. This is a magnificent score in terms of highly skewed datasets. The Isolation Forest Technique and the Local Outlier Technique were also implemented to produce fruitful results. Fu-ture research will emphasise on deducing a novel pipeline of models and techniques to optimise the scores further.

## REFERENCES

1. S. Zhang, S. Zhang, B. Wang, and T. G. Habetler, "Deep learning algorithms for bearing fault diagnostics – A com-prehensive review", IEEE Access, vol. 8, pp. 29857–29881, 2020.
2. Zhang, R.; Tao, H.; Wu, L., "Transfer learning with neural networks for bearing fault diagnosis in changing working conditions", IEEE Access 2017, 5, 14347–14357
3. C. Sobie, C. Freitas and M. Nicolai, "Simulation-driven machine learning: bearing fault classification", Mech. Syst. Signal Process. 99, 403–419 (2018).
4. B. Zhang, C. Sconyers, C. Byington, R. Patrick, M. E. Orchard and G. Vachtsevanos, "A probabilistic fault detection approach: Application to bearing fault detection", IEEE Trans. Ind. Electron., vol. 58, no. 5, pp. 2011–2018, May 2011.
5. Moldovan, D.; Cioara, T.; Anghel, I.; Salomie, I, "Machine learning for sensor-based manufacturing processes" In Pro-ceedings of the 13th IEEE International Conference on In-telligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 7–9 September 2017; pp. 147–154.
6. Munirathinam, S. and Ramadoss, S., "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.
7. Kerdprasop, K.; Kerdprasop, N., "A Data Mining Approach to Automate Fault Detection Model Development in the Semiconductor Manufacturing Process", Int. J. Mech. 2011, 5, 336–344.
8. D. Neupane and J. Seok, "Bearing Fault Detection and Diag-nosis Using Case Western Reserve University Dataset With Deep Learning Approaches: A Review", IEEE Access 2020, 8, 93155–93178.
9. Wang, J.; Feng, J.; Han, Z., "Discriminative Feature Selection Based on Imbalance SVDD for Fault Detection of Semicon-ductor Manufacturing Processes", J. Circuits Syst. Comput. 2016, 25, 1650143.
10. L. Eren, "Bearing fault detection by one-dimensional convo-lutional neural networks", Math. Probl. Eng., 2017 (2017).
11. Liu, Zhiqiang, Shravan K. Chaganti and Degang Chen. "Im-proving Time-Efficiency of Fault-Coverage Simulation for MOS Analog Circuit", IEEE Transactions on Circuits and Systems I: Regular Papers (2017).
12. Tello, Ghalia, Al-Jarrah, Y. Omar, Yoo, D. Paul, Al-Hammadi, Yousof, Muhaidat, Sami and Lee, Uihyoung, "Deep-structured machine learning model for the recognition of mixed-defect patternsin semiconductor fabrication processes", IEEE Trans-actions on Semiconductor Manufacturing, 31 (2). pp. 315-322 (2018).
13. T. Lee, K. B. Lee and C. O. Kim, "Performance of Machine Learning Algorithms for Class-Imbalanced Process Fault De -tection Problems", IEEE Trans. Semicond. Manuf. 2016, 29, 436–445.
14. X. Chen, B. Zhang and D. Gao, "Bearing fault diagnosis based on multi-scale CNN and LSTM model", J Intell Manuf (2020).
15. Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and
16. W. Philip Kegelmeyer, "SMOTE: synthetic minority over-sampling technique", J. Artif. Int. Res. 16, 1 (June 2002), 321-357.
17. 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.

## AUTHORS PROFILE

**Nikhita Mishra,** School of Computer Science and Engineering**,** Vellore Institute of Technology, Vellore

**Janhvi Mehta,** School of Computer Science and Engineering, Vellore Institute of Technology, Vellore

**Ipshitta Chaturvedi**, School of Computer Science and Engineering Vellore Institute of Technology, Vellore