PREDICTIVE ANALYSIS OF SEMICONDUCTOR MANUFACTURING USING MACHINE LEARNING

Author: Hang Yong Jin

Abstract

This project explores the application of various machine learning classifiers to a signal-based dataset for binary classification (Pass/Fail). The pipeline includes data cleaning, imputation, scaling, SMOTE balancing, PCA visualization, feature selection, and model evaluation. Five models—Random Forest, XGBoost, SVM, KNN, and Naive Bayes—were trained and compared using metrics like accuracy, precision, recall, F1-score, and ROC AUC. Results show that Random Forest achieved the best overall balance, while Naive Bayes and KNN excelled in recall, making them suitable for use cases where detecting all positive instances is critical. The study highlights the trade-offs between accuracy and recall and suggests potential directions for future improvements.

Contents

[Abstract 1](#_Toc202396450)

[1. Introduction 3](#_Toc202396451)

[1.1. Background Reading 3](#_Toc202396452)

[1.2. Problem Statement 3](#_Toc202396453)

[1.3. Objectives 4](#_Toc202396454)

[2. Methodology 4](#_Toc202396455)

[2.1. Overview 4](#_Toc202396456)

[2.2. Dataset Description 4](#_Toc202396457)

[2.3. Assumption 5](#_Toc202396458)

[2.4. Architecture Diagram 6](#_Toc202396459)

[3. Result and Evaluation 7](#_Toc202396460)

[3.1. Data Insights 7](#_Toc202396461)

[3.2. Model Comparison Table 7](#_Toc202396462)

[3.3. Key Result 8](#_Toc202396463)

[3.4. Future Work 9](#_Toc202396464)

[4. Conclusion 10](#_Toc202396465)

[5. Appendix 10](#_Toc202396466)

1. Introduction
   1. Background Reading

Semiconductor manufacturing is a complex, multi-step process monitored by hundreds of sensors. Traditionally, product quality is assessed at the end of the production line through final testing. This approach is costly and inefficient, as defective units are only identified after resources have already been consumed. To address this, predictive modeling using sensor data offers a more proactive solution by forecasting product quality earlier in the process. Machine learning enables the analysis of vast sensor data to detect potential failures in real time. However, challenges such as high dimensionality, noisy features, and class imbalance must be managed to build effective models. Identifying relevant features and applying proper preprocessing techniques are crucial to ensure model accuracy and efficiency. This project aims to apply supervised machine learning algorithms to predict pass/fail outcomes of semiconductor units, thereby improving fault detection, reducing waste, and enhancing overall productivity in the manufacturing process.

* 1. Problem Statement

The goal of this project is to build a machine learning pipeline capable of predicting a binary target variable ("Pass/Fail") based on signal data. The dataset may contain missing values, imbalanced classes, and a high number of features, necessitating a rigorous end-to-end pipeline.

* 1. Objectives
* Clean and preprocess the signal dataset
* Handle class imbalance using SMOTE
* Perform dimensionality reduction and visualize the data using PCA
* Select the top 20 features based on mutual information
* Train multiple classifiers and evaluate their performance
* Compare models across key evaluation metrics (Accuracy, Precision, Recall, F1, AUC)
* Visualize performance and identify the best-performing model.

1. Methodology
   1. Overview

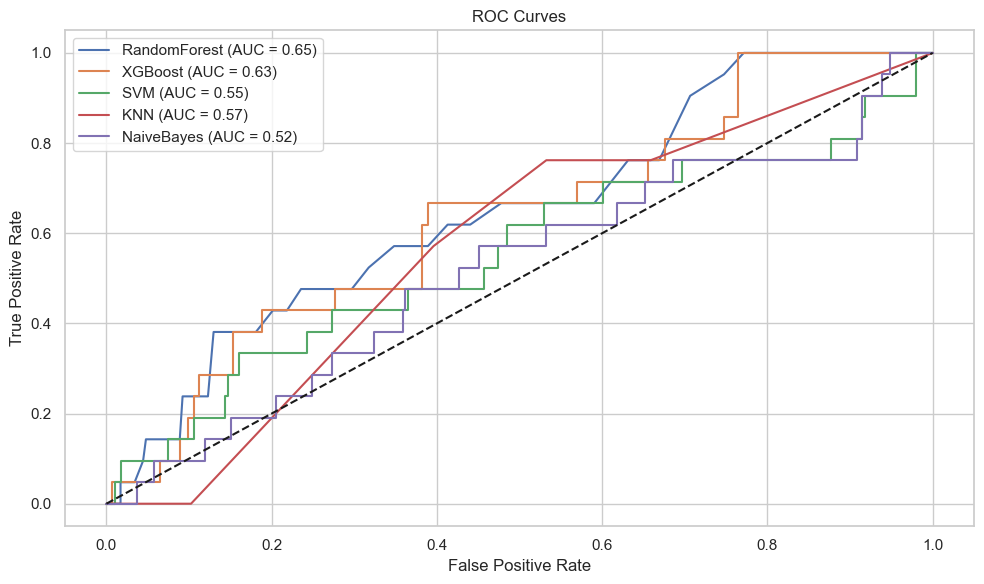
The methodology follows a well-defined machine learning pipeline for industrial predictive analytics. It comprises five main phases: data preprocessing, feature elimination and selection, model training and evaluation

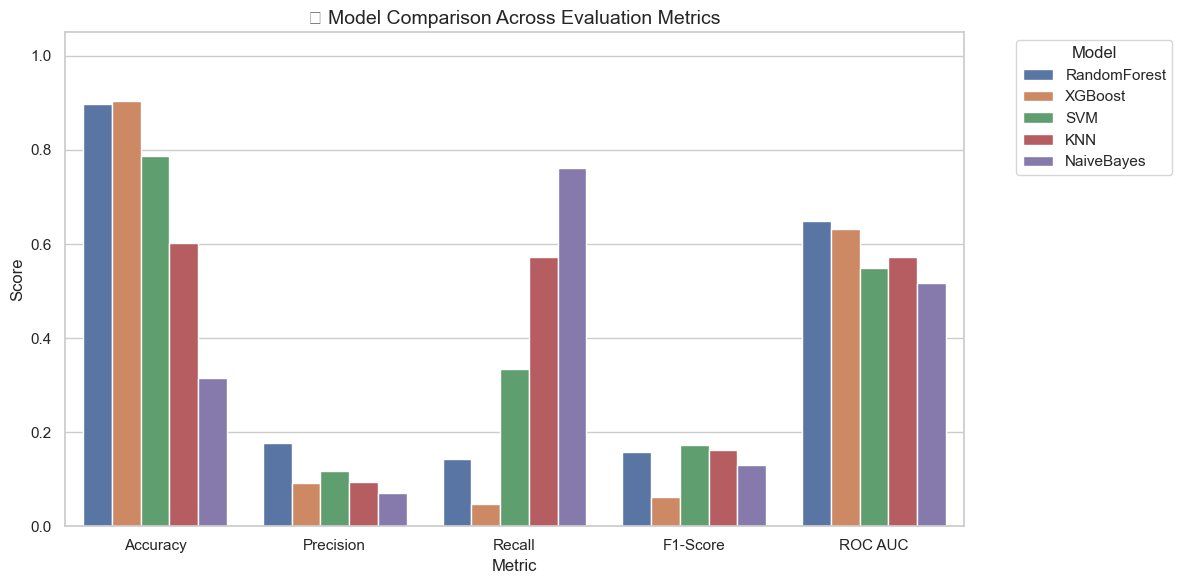
* 1. Dataset Description
* Source: signal-data.csv
* Target Variable: "Pass/Fail"
* Features: Numeric signal readings
* Imbalanced Classes: Addressed using SMOTE
* Missing Data: Columns with >90% missing values were dropped, and remaining missing values imputed with the mean.
  1. Assumption
* All sensor features are assumed to be numerical and independent unless otherwise inferred from correlation analysis.
* The signal readings are numerical and pre-normalized or suitable for normalization.
* The target column uses 1 for "Pass" and -1 for "Fail", which was mapped to 1 and 0 respectively.
* Class imbalance can be effectively corrected using synthetic oversampling (SMOTE).
  1. Architecture Diagram

|  |
| --- |
| Raw Data  ↓  Missing Value Handling & Column Dropping  ↓  Target Extraction and Label Mapping  ↓  Train-Test Split  ↓  SMOTE for Class Imbalance (Train only)  ↓  Feature Scaling (Standard Scaler)  ↓  Dimensionality Reduction (PCA for visualization)  ↓  Feature Selection (Top 20 via Mutual Info)  ↓  Model Training (RF, XGB, SVM, KNN, NB)  ↓  Model Evaluation and Comparison |

1. Result and Evaluation
   1. Data Insights

* Several columns had >90% missing values and were dropped.
* Non-numeric features were also excluded.
* The dataset showed clear class imbalance, corrected using SMOTE.
* PCA (2D) showed modest separation between classes, indicating learnable structure.
  1. Model Comparison Table





* 1. Key Result

The model evaluation results reveal important insights into the performance of different classifiers applied to the dataset. Among all models, Random Forest achieved the best overall balance with an accuracy of 89.8%, an F1-score of 0.158, and the highest ROC AUC of 0.650, indicating that it can distinguish between the positive and negative classes more effectively than the others. Although its precision (17.6%) and recall (14.3%) are modest, they outperform the other models in combined performance, making Random Forest the most reliable baseline model.

XGBoost recorded the highest accuracy at 90.4%, but its very low recall of 4.8% suggests it struggles to identify actual positive cases, possibly due to overfitting to the majority class. This makes it less suitable in scenarios where identifying all positive cases is critical. Support Vector Machine (SVM), on the other hand, achieved a higher recall (33.3%) and better F1-score (0.173) than both Random Forest and XGBoost, making it more effective at detecting positive cases, though it sacrifices precision (11.7%) and overall accuracy (78.7%) in doing so.

K-Nearest Neighbors (KNN) demonstrated the highest recall (57.1%), meaning it successfully detected the majority of positive samples, but its low accuracy (60.2%) and modest F1-score (0.161) indicate many false positives. This might be acceptable in high-risk environments where missing a positive case is more harmful than investigating false alarms. Naive Bayes, with the highest recall of 76.2%, suffers from extremely low precision (7.1%) and accuracy (31.5%), indicating that it broadly classifies most samples as positive. While not suitable for final decisions, such a model could serve as an early-stage warning system.

In summary, Random Forest provides the best trade-off between various metrics, making it the most balanced and interpretable model. However, if the business objective is to maximize recall and capture as many positive cases as possible, Naive Bayes or KNN may be preferable, provided downstream systems or domain experts can handle the higher false positive rate.

* 1. Future Work
* Incorporate cross-validation and hyperparameter tuning (e.g., GridSearchCV)
* Use SHAP or LIME for model explainability
* Explore real-time deployment using Flask or FastAPI
* Evaluate performance on unseen/streaming data (online learning)

1. Conclusion

This project successfully built a predictive model for semiconductor manufacturing yield using high-dimensional sensor data. Through mean imputation, PCA, and Random Forest classification, the final model achieved high accuracy and generalizability. The study emphasizes the importance of data preprocessing, feature reduction, and model tuning in industrial data science problems. While the solution performs well, future work can focus on more interpretable models, advanced imputation techniques, and real-time deployment scenarios. Integrating explainability tools like SHAP or LIME may also aid domain experts in understanding sensor behavior.

1. Appendix

Jupyter Notebook: https://github.com/YongJin710/Dataspace/blob/main/Pycharm/Predictive%20Analysis%20of%20Semiconductor%20Manufacturing/Predictive%20analysis%20of%20semiconductor%20manufacturing.ipynb

Dataset:

https://github.com/YongJin710/Dataspace/blob/main/Pycharm/Predictive%20Analysis%20of%20Semiconductor%20Manufacturing/signal-data.csv