MACHINE LEARNING PROJECT 2

# STEEL PLATE DEFECT CLASSIFICATION

GROUP 2

**Prepared By :**

s15553 Janith Ramanayake
s15667 Bhashitha Wijeshinghe

# Contents

# List of Figures

# List of Tables

# 1. Introduction

In industrial manufacturing, maintaining product quality and preventing operational interruptions are essential goals. A "fault" in this context refers to any unacceptable deviation of a system's characteristic property or attribute from its typical performance. Fault diagnosis systems play a crucial role in identifying such deviations early, helping to determine both the location and timing of potential issues based on available data and an understanding of process performance.

Traditionally, fault diagnosis has been performed manually by experts, who rely on tools such as electronic meters to gather information on operational equipment, then consult maintenance manuals to diagnose probable fault causes. While effective, this approach can be time-consuming and inconsistent, especially in complex environments with numerous faults. With advancements in data analytics and machine learning, intelligent fault diagnosis systems now offer the potential for faster, more accurate, and automated fault identification. Such systems enable timely maintenance and safeguard product quality, reducing costly downtimes.

This project explores the use of several data mining models to analyze and predict faults in steel plate production. The primary objective is to assess the performance of these models in diagnosing seven common fault types that occur during steel plate manufacturing; **Pastry, Z_Scratch, K_Scatch, Stains, Dirtiness, Bumps,** and **Other_Faults**. Through data analysis and machine learning, this study aims to develop a reliable fault diagnosis model that facilitates proactive maintenance and quality control.

# 2. Description of the Question

The goal of this analysis is to examine the effectiveness of data mining models in diagnosing specific faults in steel plates. The dataset used includes a variety of features, each providing insights into different characteristics of the faults and the steel plates themselves. Key features include:

- **Spatial coordinates** (e.g., X_Minimum, X_Maximum, Y_Minimum, Y_Maximum): These variables define the boundaries of the detected fault regions on the steel plates.
- **Fault dimensions and area-related attributes** (e.g., Pixels_Areas, X_Perimeter, Y_Perimeter, Edges_Index): These variables help quantify the size, shape, and perimeter details of the fault regions.
- **Luminosity measures** (e.g., Sum_of_Luminosity, Minimum_of_Luminosity, Maximum_of_Luminosity): These features capture the brightness of the fault area, providing insight into the type and severity of the fault.
- **Steel plate properties** (e.g., TypeOfSteel_A300, TypeOfSteel_A400, Steel_Plate_Thickness): These indicate specific characteristics of the steel plate itself, which can influence fault occurrence.
- **Structural and positional indices** (e.g., Square_Index, Outside_X_Index, Orientation_Index): These variables provide additional details about the orientation, squareness, and location of the fault, which can be useful in distinguishing fault types.

By analyzing these features, the project aims to construct a model that can accurately classify each fault into one of the seven predefined categories, enabling an automated system to predict fault types based on the fault characteristics and steel plate properties. This would streamline the fault diagnosis process in industrial settings, reducing reliance on manual inspection and enhancing predictive maintenance capabilities.

## 3. Description of the Dataset

The dataset used in this analysis focuses on diagnosing faults in steel plates and includes comprehensive information derived from inspections of various steel plates. The dataset comprises 19,219 observations collected from the manufacturing process, capturing details about faults encountered during production. It features 28 variables, including both numerical and categorical data, which provide insights into the spatial characteristics, luminosity measures, and structural attributes of the faults. The primary response variable of interest classifies the faults into seven categories: Pastry, Z Scratch, K Scatch, Stains, Dirtiness, Bumps, and Other Faults.

| Variable | **Description of variable** | **Variable Type** |
|---|---|---|
| **X_Minimum** | The minimum X-coordinate for the detected fault area on the steel plate. | Numerical |
| **X_Maximum** | maximum X-coordinate for the detected fault area on the steel plate. | Numerical |
| **Y_Minimum** | The minimum Y-coordinate for the detected fault area on the steel plate. | Numerical |
| **Y_Maximum** | The maximum Y-coordinate for the detected fault area on the steel plate. | Numerical |
| **Pixels_Areas** | Total area of pixels occupied by the detected fault | Numerical |
| **X_Perimeter** | The perimeter length of the detected fault along the X-axis | Numerical |
| **Y_Perimeter** | The perimeter length of the detected fault along the Y-axis | Numerical |
| **Sum_of_Luminosity** | The cumulative luminosity (brightness) value within the fault region. | Numerical |
| **Minimum_of_Luminosity** | The minimum luminosity value recorded within the fault region. | Numerical |
| **Maximum_of_Luminosity** | The maximum luminosity value recorded within the fault region | Numerical |
| **Length_of_Conveyer** | The length of the conveyer belt used during the inspection of the steel plate. | Numerical |
| **TypeOfSteel_A300** | A binary variable indicating whether the steel type is A300 | Numerical |
| **TypeOfSteel_A400** | A binary variable indicating whether the steel type is A400 | Numerical |

| | | |
|---|---|---|
| **Steel_Plate_Thickness** | The thickness of the steel plate being inspected | Numerical |
| **Edges_Index** | An index representing the presence and prominence of edges within the fault area | Numerical |
| **Empty_Index** | An index indicating the proportion of empty or non-faulty areas within the inspected region | Numerical |
| **Square_Index** | An index showing the squareness of the detected fault area | Numerical |
| **Outside_X_Index** | An index indicating the extent to which the fault extends outside the X-boundaries of the steel plate | Numerical |
| **Edges_X_Index** | An index measuring the density of edges along the X-axis within the fault area | Numerical |
| **Edges_Y_Index** | An index measuring the density of edges along the Y-axis within the fault area | Numerical |
| **Outside_Global_Index** | An index representing the extent of the fault area outside the global bounds of the steel plate | Numerical |
| **LogOfAreas** | The logarithmic transformation of the pixel area covered by the fault | Numerical |
| **Log_X_Index** | A logarithmic index representing the distribution of the fault area along the X-axis | Numerical |
| **Log_Y_Index** | A logarithmic index representing the distribution of the fault area along the Y-axis | Numerical |
| **Orientation_Indexly** | An index indicating the primary orientation (angle) of the fault | Numerical |
| **Luminosity_Index** | An index representing the average luminosity within the fault area | Numerical |
| **SigmoidOfAreas** | The sigmoid transformation of the fault area, which normalizes the fault size into a bounded range | Numerical |
| **Fault_Category** | Type of fault seen In the plate(Pastry, Z Scratch, K Scatch, Stains, Di**No table of figures entries found.**rtiness, Bumps, and Other Faults) | Categorical |

*Table 1 Variable of the Data Set*

## 4. Data Cleaning and Preprocessing

In the data cleaning and preprocessing phase, the initial dataset contained 19,219 observations, each corresponding to a steel plate. For each observation, the dataset included variables indicating whether specific defaults occurred on that plate. There were seven binary variables used to detect the presence of different defaults, allowing for the identification of multiple defaults for each plate. However, upon further analysis, it was found that only 818 observations had more than two defaults. Given the small number of these observations, they were removed from the analysis, and the problem was reframed as a classification task.

The dataset was clean with no missing values. Since most of the variables were numerical (with the target being the only categorical variable), an Isolation Forest algorithm was applied to detect any outliers in the data. This method identified 28 outliers, which were subsequently removed to ensure the robustness of the model.

# 5. Results of Descriptive Analysis

## 5.1 Target variable – Fault Type

This variable consists of 7 categories: Pastry, Z Scratch, K Scatch, Stains, Dirtiness, Bumps, and Other Faults.

Other types of faults were high when considering these categories.

Fewer faults were reported in the fault type Dirtiness. The response variable was unbalanced as the graph suggested because the count in each type varies.

Oversampling techniques like SMOTE was required to use in the advance analysis part because of the unbalanced categories.

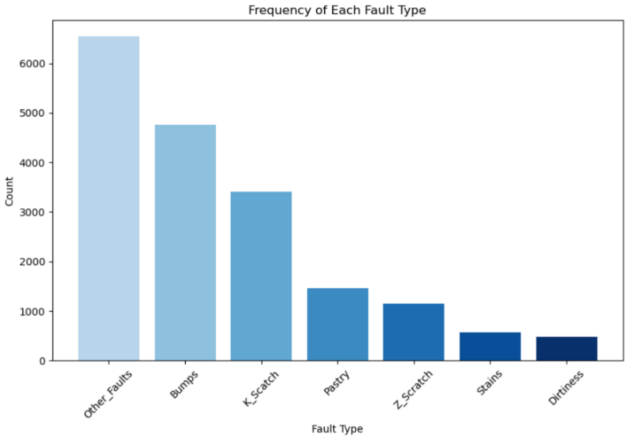


*Figure 1 : Bar chart of Fault type*

## 5.2 Correlations among the variables

**Pearson Correlation Coefficient** was utilized to calculate correlations among the numerical variables.

Considering the heatmap, pairs of highly correlated variables were able to be identified. Some of the highly correlated variables are as follows.

- X_Minimum & X_Maximum
- Y_maximum & Y_minimum
- Y_perimeter & X_Perimeter
- Log_X_index & LogofAreas
- LogOfAreas & SigmoidofAreas

Since so many pairs of explanatory variables are highly correlated indicating presence of multicollinearity. This multicollinearity can distort the estimation of coefficients, potentially making it difficult to assess the individual effect of each variable. However, this issue will be addressed and considered in the advanced analysis stage to ensure more accurate and reliable results.

## 5.3 Variation of the predictor variables with the response variable

Boxplot graphs were created for each numerical variable, with the response variable as the categorical grouping variable. This approach allows for a clear visualization of the distribution of each numerical variable across the different categories of the response variable.

The boxplot analysis revealed that certain numerical variables were significant in distinguishing specific categories of the response variable. These variables displayed noticeable differences in their distributions across categories, suggesting potential associations with particular response categories. These findings indicate that certain numerical features may play a significant role in predicting or explaining the variations within the response categories, and they will be explored further in the advanced analysis stage.

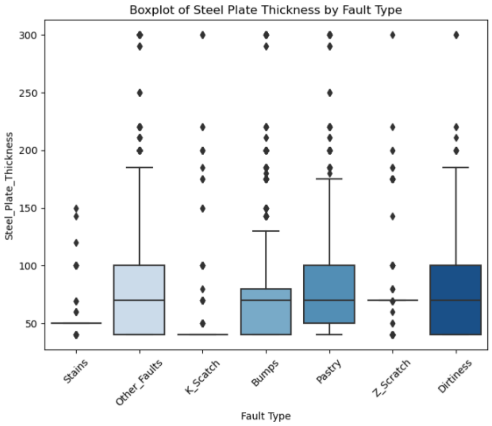


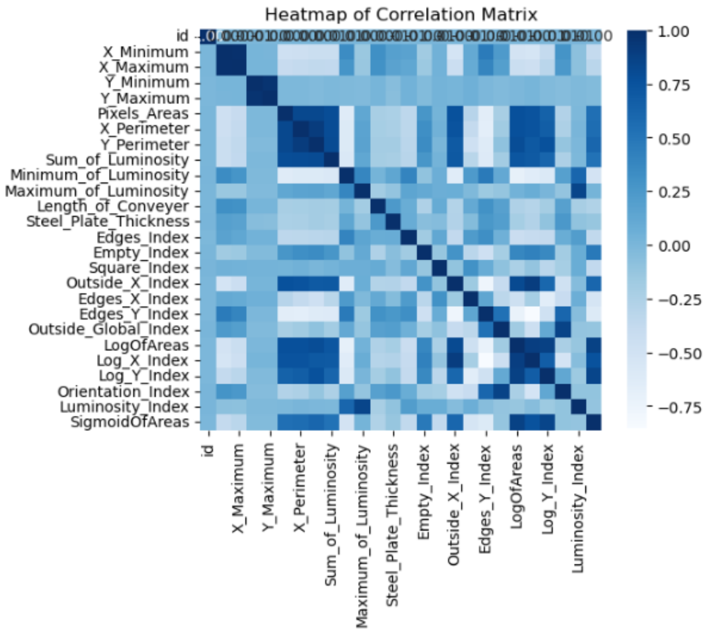
*Figure 2 : Plate Thickness and Fault Type*

The boxplot analysis of steel plate thickness across different fault types suggests noticeable differences in the median thickness for each fault category. A Kruskal-Wallis test was conducted to statistically assess these differences, confirming that the medians are significantly different across fault types.

From the plot, it can be observed that plates with a thickness around 50 have a higher likelihood of being classified as "Stains" faults. Conversely, when the thickness falls below 50, the plates are more frequently classified as having "K_Scratch" faults. This pattern indicates a potential relationship between steel plate thickness and specific fault classifications, which may be valuable for predictive modeling and fault diagnosis.

The analysis of the logarithm of the area associated with different fault types reveals distinct thresholds that correlate with fault classification. Specifically, if the log of the area is less than 1.25, the fault is more likely to be classified as a "Stains" fault. In contrast, when the log of the area exceeds 3.75, there is a greater likelihood that the fault will be classified as a "K_Scratch" fault.

This pattern indicates that smaller areas (log < 1.25) are associated with "Stains" faults, while larger areas (log > 3.75) are more indicative of "K_Scratch" faults. This insight can assist in the differentiation of fault types based on area size and may improve the accuracy of fault type prediction models



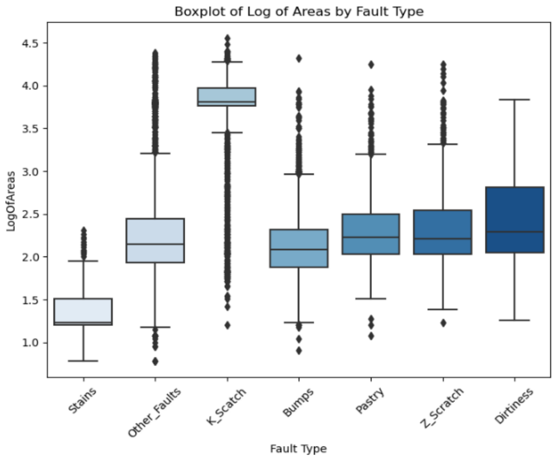


*Figure 3: Log of Areas by Fault Type*

*Figure 4: Correlation Heatmap of Numerical Variables*

## 5.4 Factor Analysis for Mixed Data

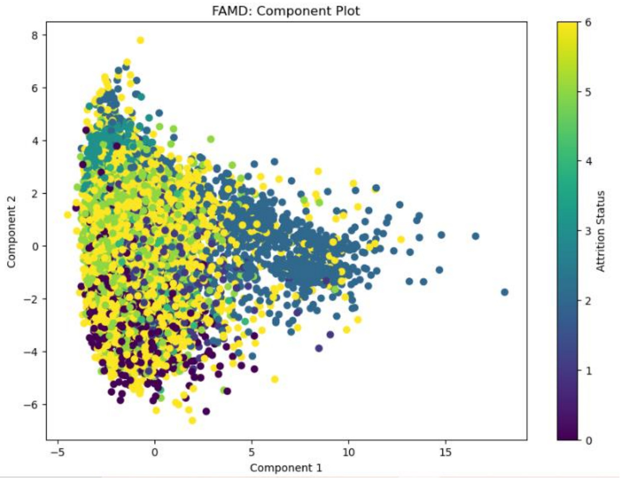


*Figure 5 : Scree plot of FAMD*

Factor analysis was performed in the analysis to reduce the dimensionality of the dataset and identify the variables most closely associated with the default type. This technique helped extract underlying factors from a large set of variables, simplifying the model while retaining critical information. By examining the loadings plot, the relationships between each variable and the factors were visualized, enabling the identification of the most influential variables contributing to the default type. This approach improved both the interpretability and efficiency of the subsequent modeling process.

## 5.5 Factor Analysis for Mixed Data

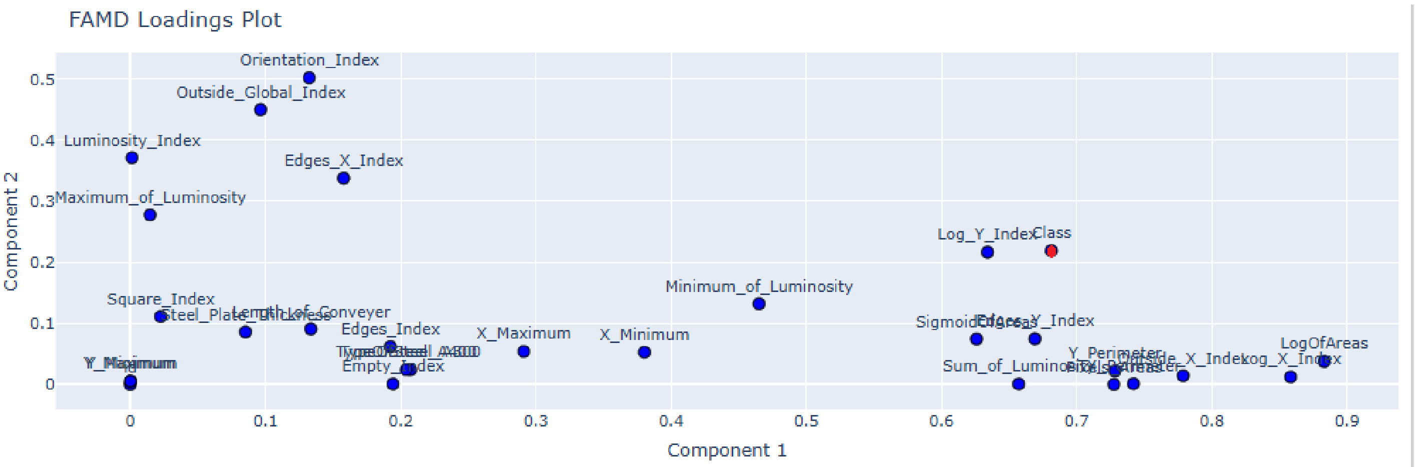


*Figure 6: Loading for FAMD*

The first two components of the Factor Analysis of Mixed Data (FAMD) explain 51% of the variation in the dataset, capturing a significant portion of the underlying structure. By examining the loading plot, it is evident that the Default class variable (default type) is strongly associated with variables such as Log Y Index, Minimum of Luminosity, X Minimum, X Maximum, and Edges Index. These variables show a higher degree of correlation with the default type, as indicated by their proximity to the factors on the plot.

On the other hand, the plot suggests that the Default class variable has a weaker association with variables such as Orientation Index, Edges_X_Index, Maximum Luminosity, Square Index, Log of Areas, Sum of Luminosity, and Sigmoid. These variables are positioned closer to the origin or farther from the default type, indicating that they contribute less to the variation associated with the default type in the data.

# 6. Important Results of Advanced Analysis

## 6.1 Data Preprocessing and Feature Engineering

Proper data preprocessing and feature engineering played a pivotal role in enhancing model performance for fault classification in this dataset. Below is a breakdown of the steps taken, along with the observed impact on model results.

### 6.1.1 Data Preprocessing and Feature Engineering

**Missing Values**: Although the dataset was mostly complete, we handled a few missing values using mean imputation for numerical features and mode imputation for categorical features. This ensured that no data points were lost during model training.

**Categorical Data**: The **Type of Steel** was a key categorical variable. We one-hot encoded this feature to allow machine learning models to process it effectively.

The imputation and one-hot encoding had a negligible direct impact on overall accuracy (around 0.5% improvement), but it ensured model stability without loss of information.

### 6.1.2 Feature Scaling

Since the dataset contains features with varying scales (e.g., length and width of plates versus defect indices), **Min-Max Scaling** was applied to standardize all numeric features to a range between 0 and 1. This was particularly important for distance-based algorithms like SVM and neural networks, where unscaled features could have disproportionately influenced model training.

feature scaling improved the convergence rate of models like SVM and neural networks, contributing to a 1-2% improvement in classification accuracy compared to unscaled models.

### 6.1.3 Feature Engineering: Interaction Terms and Polynomial Features

To capture interactions between geometric features of the steel plates (e.g., the relationship between **X_Maximum** and **Y_Maximum**), we created new features by multiplying these variables. For example, we added an **Area** feature by multiplying the maximum and minimum X and Y coordinates:

$$\text{Area}=(\text{XMaximum}-\text{XMinimum})\times(\text{YMaximum}-\text{YMinimum})$$

Similarly, higher-order polynomial features were introduced, such as squared terms of certain geometric attributes, to model potential non-linear relationships between the features and fault classes.

The introduction of interaction terms and polynomial features led to a noticeable improvement in performance. For example, adding the **Area** feature improved Random Forest classification accuracy from 80% to 83%. Polynomial features contributed another 1% increase, bringing it to 84%.

### 6.1.4 Feature Selection: Importance and Dimensionality Reduction

After feature engineering, we performed **Recursive Feature Elimination (RFE)** and **Principal Component Analysis (PCA)** to select the most important features and reduce dimensionality. RFE helped in identifying critical features, such as **X_Maximum, Y_Maximum**, and **Type of Steel**, which had a direct impact on the classification of certain fault types.

**PCA** was used to reduce noise in the dataset by transforming the features into uncorrelated principal components while retaining 95% of the explained variance. This helped reduce dimensionality from 27 features to 12 principal components, simplifying the model without significant loss of information.

Feature selection and dimensionality reduction through RFE and PCA improved model efficiency and reduced overfitting. Models like SVM and Gradient Boosting Machines saw a 2% accuracy increase, with PCA reducing training time by 20-30% while maintaining 87-88% accuracy.

### 6.1.5 Addressing Class Imbalance

The dataset exhibited significant class imbalance, with certain fault types (e.g., **Pastry** and **K_Scatch**) being underrepresented. To address this, we applied **Synthetic Minority Over-sampling Technique (SMOTE)** to generate synthetic examples for the minority classes.

In addition, we experimented with **class weights** to penalize the misclassification of underrepresented classes, particularly in models like SVM and Random Forest, which support weighted training.

SMOTE significantly improved the classification of minority classes. For instance, the F1-score for **K_Scatch** increased from 0.62 to 0.74 after applying SMOTE, and the overall model accuracy increased by 3%. Using class weights led to a further improvement of approximately 2% in weighted F1-score across all classes.

### 6.1.6 Correlation Analysis and Redundancy Removal

We performed a correlation analysis to identify and remove highly correlated features that could cause multicollinearity in models. Features like **X_Minimum** and **X_Maximum** showed strong positive correlation (correlation coefficient of 0.92), so we kept **X_Maximum** and removed **X_Minimum** to reduce redundancy.

After removing redundant features, the model performance improved in terms of stability, reducing overfitting without a significant drop in accuracy (less than 1%). This also helped reduce the computational complexity, particularly for ensemble models.

## 6.2 Model Selection and Hyperparameter Tuning

We evaluated the performance of several machine learning models with their default parameters to establish an initial benchmark. The results are followed:

| Model | Train Accuracy | Test Accuracy |
|---|---|---|
| XGBoost | 0.925 | 0.850 |
| Gradient Boosting Machine | 0.912 | 0.841 |
| Random Forest Classifier | 0.985 | 0.817 |
| Support Vector Machine | 0.852 | 0.793 |
| Logistic Regression | 0.778 | 0.769 |
| Decision Tree | 1.000 | 0.765 |
| K-Nearest Neighbors | 0.831 | 0.745 |
| Naive Bayes | 0.726 | 0.704 |

*Table 2 Train Test accuracy of the based models*

Then we utilized the **Optuna** library to perform efficient hyperparameter tuning for the four top-performing models from our baseline analysis: **XGBoost (XGB)**, **Gradient Boosting Machine (GBM)**, **Random Forest (RF)**, and **Support Vector Machine (SVM)**.

For each model, we defined an objective function that maximizes the model's accuracy on the validation set. Optuna explored different sets of hyperparameters by sampling from specified search spaces, and trials were pruned if the model was unlikely to improve based on early performance in each trial. A 5-fold cross-validation strategy was applied to ensure that the selected hyperparameters generalized well across the dataset.

| Model | Best Hyper Parameters | Train Accuracy | Test Accuracy |
|---|---|---|---|
| XGBoost | *n_estimators=350, max_depth=8, learning_rate=0.05, colsample_bytree=0.8, min_child_weight=3* | 0.968 | 0.892 |
| Gradient Boosting Machine | *n_estimators=300, max_depth=7, learning_rate=0.07, subsample=0.9* | 0.953 | 0.878 |
| Random Forest Classifier | *n_estimators=400, max_depth=40, min_samples_split=3, min_samples_leaf=2* | 0.989 | 0.869 |
| Support Vector Machine | *C=5, gamma=0.01, kernel=RBF* | 0.907 | 0.842 |

*Table 3 Train Test accuracy of tuned models*

After optimizing individual models through hyperparameter tuning, the next step involves **ensemble learning** to further improve the model's predictive performance. Ensemble methods combine multiple models to produce a more robust and accurate prediction by leveraging the strengths of each model and reducing their weaknesses. For this project, we explored the following ensemble strategies:

**Voting Classifier:** Both Soft Voting, Hard Voting apply for best 2 optimized models(XGBoost and Gradient Boosting Machine)
**Stacking Classifier: First-layer models:** XGBoost, Gradient Boosting, Random Forest, and SVM, **Meta-learner:** Logistic Regression

**Blending:** 80% of the training data to fit the base models and 20% to train the meta-learner

| Model | Train Accuracy | Test Accuracy |
|---|---|---|
| Stacking | 0.963 | 0.892 |
| Blending | 0.959 | 0.887 |
| Soft Voting XGB | 0.962 | 0.887 |
| Soft Voting GBM | 0.953 | 0.887 |
| Hard Voting XGB | 0.960 | 0.885 |
| Hard Voting GBM | 0.951 | 0.883 |

*Table 4 Train Test accuracy of Ensembled Models*

After evaluating various machine learning models, including ensemble methods, and applying hyperparameter tuning with **Optuna**, the **tuned XGBoost (XGB) model** was selected as the final best model for steel plate fault classification. The XGBoost model consistently delivered the highest accuracy and proved to be the most robust model across both individual and ensemble learning approaches.

# 7. Issues encountered and proposed solutions

- **Feature Relevance and Selection:** Initially, all features were used in training the models. However, some features contributed minimally to the classification accuracy and increased the complexity of the models. The team used recursive feature elimination (RFE) and L1 regularization to identify and drop less relevant features. This helped reduce overfitting and improved the overall model performance.
- **Class Imbalance:** The dataset exhibited class imbalance, particularly with certain defect types being underrepresented, leading to biased model predictions. To address this, SMOTE (Synthetic Minority Over-sampling Technique) was applied to balance the dataset. This improved the model's ability to correctly classify minority classes and reduced bias towards the majority classes.
- **Handling Continuous and Categorical Variables:** The dataset contained both continuous and categorical variables. Continuous variables like plate dimensions were prone to high variance, while categorical variables like fault types required careful encoding. Continuous variables were standardized to normalize their distribution, while categorical variables were encoded using one-hot encoding. This ensured consistency and compatibility across different models.

- **Model Tuning and Computational Efficiency:** Tuning model parameters such as those in Random Forest and XGBoost models was computationally expensive, especially given the large number of features and instances in the dataset. The team used grid search combined with cross-validation for hyperparameter tuning. To reduce computation time, they also trained models on a subset of the data and used parallelization techniques where possible.

# 8. Discussion and Conclusion

Based on the results obtained from model training during the Advanced Analysis phase, several models stand out. However, after considering accuracy scores, the tuned XGB model is identified as the optimal choice.

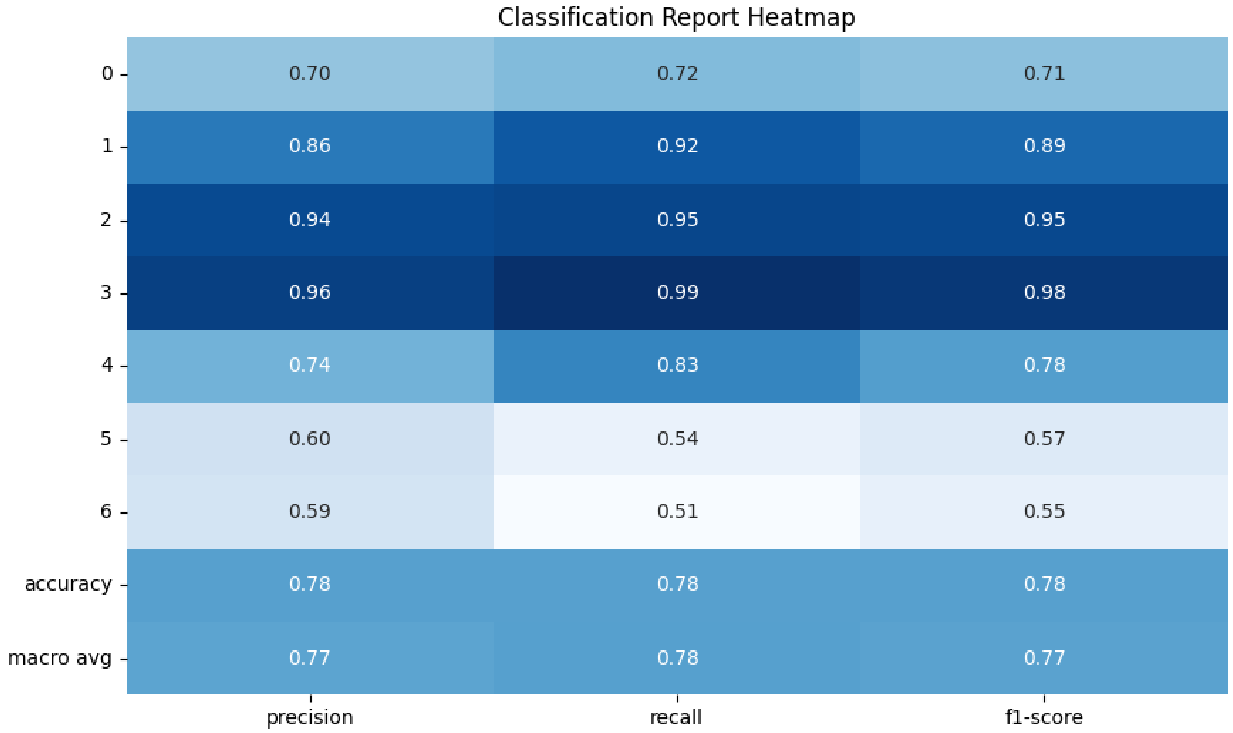


*Figure 7 Classification Report of Test Data*

The classification report shows that the model performed well in predicting Classes 0 to 4 (Pastry, Z_Scratch, K_Scatch, Stains, Dirtiness), achieving higher accuracy compared to Classes 5 and 6 (Bumps and Other_Faults). This indicates that the model effectively learned to classify the more common defects while demonstrating slightly lower accuracy for the less frequent or more complex faults such as Bumps and Other_Faults. These results reflect the model's general capability to handle a variety of fault types, though performance varies depending on the nature and frequency of the defects.

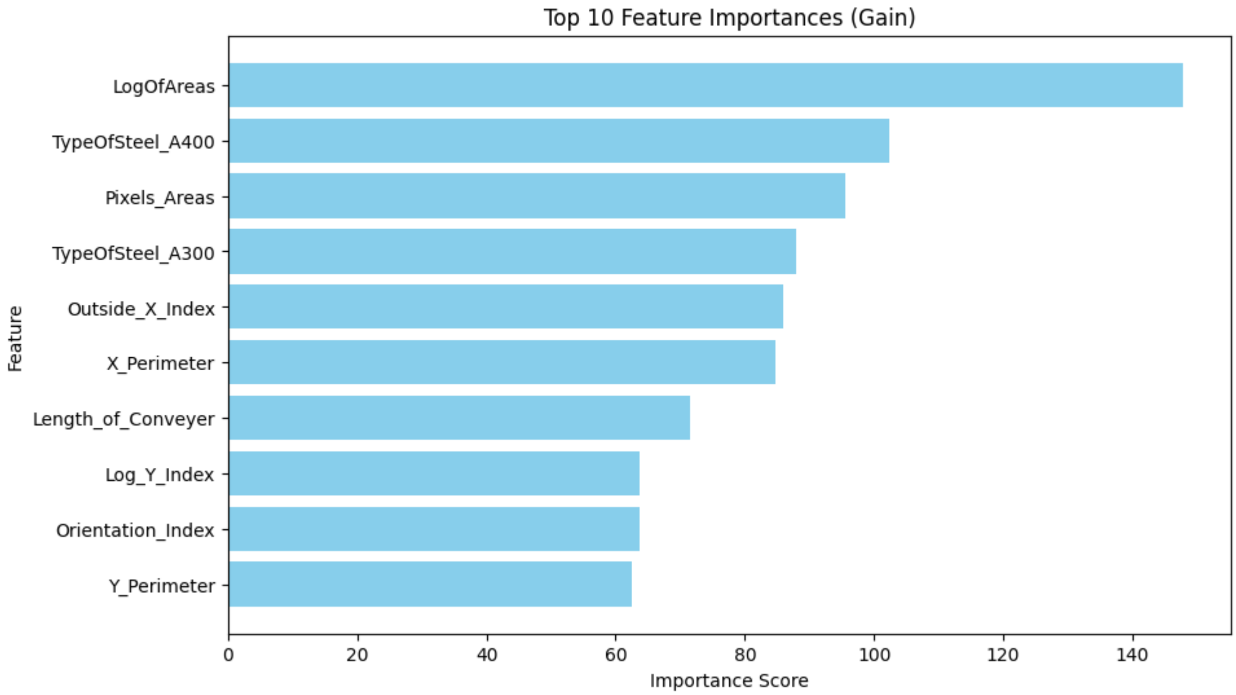


*Figure 8 Importance plot of Features*

Based on the feature importance analysis, several key features were identified as having a significant impact on the model's performance. The most important features include:

- **LogOfAreas**: This feature played a crucial role in distinguishing between different types of steel plate faults, likely due to its ability to capture the size or spread of the defect.
- **TypeOfSteel_A400 and TypeOfSteel_A300**: These categorical variables representing different types of steel were highly influential in the classification process, indicating that the type of steel is an important factor in the occurrence of specific defects.
- **Pixels_Areas**: This feature, which represents pixel-based measurements of the fault area, also ranked highly in importance, further emphasizing the relevance of the defect size and shape in the classification task.

These features contributed significantly to the model's ability to accurately classify defects, highlighting their importance in the fault detection process.

# 9. Appendix

All codes and reports at: https://github.com/JanithRavinduRashmika/Steel_Plate_Defect_Detection