Steel Defect Prediction with Ensemble modeling

20191612 윤기웅
20191588 박성진

서강대학교 컴퓨터공학과
(발표자 연락처**:** rldnddbs@naver.com **)**

ABSTRACT

We participated in the Kaggle competition "Steel Defect Prediction" [1]. The task is to predict the probability of seven defect types of steel plate. For feature engineering, we use feature extraction, creating derived features, clustering, and log scaling. For modeling, we ensembled four classification models XGBoost, CatBoost, LightGBM, and HistGradientBoostingClassifier. Additionally, we applied different weights to each target. This ensemble model showed great performance, placing in the top 4% of total 2199 teams. In this paper, we will explain the steps involved in constructing a machine learning model for this task.

[Figure 1: Final Result]

Keywords: Feature Engineering, Ensemble Learning, Steel Defect Detection, Multi-label Classification, Kaggle Competition

1. Introduction

Steel is important and it plays a vital role in our daily lives. Korean enterprise POSCO, mainly focusing on the steel industry, is one of the biggest steel companies in the world. Also the company mainly led the growth of Korea economy in the last decades [8]. In 2023, POSCO began making electrical steel plate. Recently, the demand for electrical steel plates has been elevated due to the rise in electric vehicle production, which shows the importance of steel plates quality.

As a result, detecting defects in steel plates and verifying their quality has become increasingly significant. In this study, we aim to identify potential defects in steel plates using machine learning. The Steel Defect Prediction competition, hosted by Kaggle, started on March 1, 2024, and ended on March 31, 2024. It was evaluated based on the ROC-AUC score, with a total of 2,199 teams participating. The task was to predict the types of defects present in given features.

2. Data Explanation

2-1. Features

The dataset includes 27 features that provide information about defects in steel plates. These features can be grouped into five categories based on their characteristics:

Coordinate: These features describe the spatial coordinates of the faults on the steel plate. This includes the minimum and maximum x-coordinates and y-coordinates. This can offer positional information of defect in each steel plate.

Size: These features express the area of the faults. The "Pixels_Areas" feature represents the fault area in pixels while "X_Perimeter" and "Y_Perimeter" measure the fault's perimeter along the x-axis and y-axis. These metrics are essential for understanding the physical size and shape.

Luminosity: These features measure the light intensity within the fault area. It includes the total luminosity and the minimum and maximum luminosity values. They help assess the brightness of the faults, which can indicate their nature and severity.

Material & Index: These features include the type of steel and the thickness of the steel plate. Additionally, index values related to the geometry and edges of the faults are included. This can provide a comprehensive description of the faults.

Other Features: This category includes features derived by applying logarithmic and sigmoid functions to the above-mentioned features, such as “LogOfAreas” and “SigmoidOfAreas.”

2-2. Targets

Seven defect types in the data that can be found in steel plates are given as targets [2]. The task is to predict the probability of each defect type. Thus, this is a multi-label classification task. These are the seven targets and its descriptions.

Pastry : Small patches or irregularities on the surface of the steel plate.

Z_Scratch : Narrow scratches on the surface of the steel plate that run parallel to the rolling direction.

K_Scratch : Narrow scratches on the surface of the steel plate that run perpendicular to the rolling direction.

Stains : Discolored or contaminated areas on the surface of the steel plate.

Dirtiness : The presence of dirt or particulate matter on the surface of the steel plate.

Bumps : Raised or protruding areas on the surface of the steep plate.

Other_Faults : Defects that are not explicitly categorized in the above.

3. Data preprocessing and Feature Engineering

Understanding and processing the given data is crucial in machine learning competition. We applied various techniques such as feature extraction, creating derived features, clustering,

and log scaling to ensure our model to learn effectively. We applied those methods while avoiding too much model complexity and overfitting.

3-1. Feature Extraction

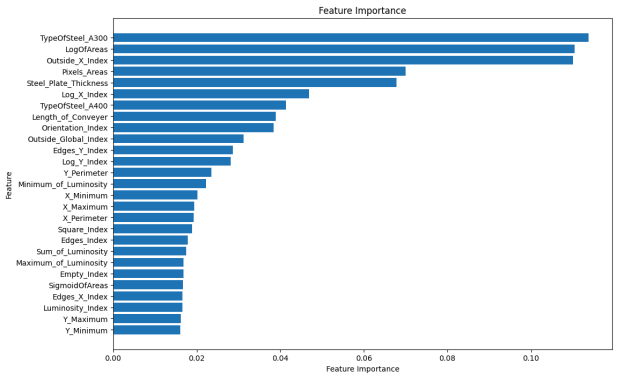
To improve the performance and simplicity of our model, we removed several features that were determined to be less important for predicting defects in steel plates. Also we removed some features that are included in derived features.

These features are removed by low feature importance in XGB. Feature importance graph is shown in Figure 2.

- Y_Minimum
- Luminosity_Index
- Edge_X_Index
- SigmoidOfAreas

These features are included in derived variables or clustered features. We eliminate these features for simplicity of the model.

- Steel_Plate_Thickness
- Sum_of_Luminosity



[Figure 2: Feature importance in XGB]

3-2. Derived features

We made new features from the existing data to provide better insights and improvement of the model performance. We describe the new features added to the dataset below.

Ratio of Length to Thickness : This feature is calculated as the Length_of_Conveyor divided by the Steel_Plate_Thickness. A longer conveyor with thin steel plates might mean too much processing which might cause more defects to the plate. In contrast, a shorter conveyor with thick steel plates might mean lower processing.

Product of X-Range and Pixels Areas : This feature is derived by multiplying the range of the X-dimension (X_Maximum - X_Minimum) with the Pixels_Areas. If this value is high, it means that there exists defects in X-dimension. It can be useful for the model to learn different patterns of K_Scratch and Z_Sractch.

Average Luminosity : This feature represents the average luminosity of the steel plate, calculated by dividing the Sum_of_Luminosity by the Pixels_Areas. Luminosity can be an indicator of surface properties and potential defects. Since given features are the sum of the luminosity, it can be affected by the area of defect. So we normalize the total luminosity by the defects area.

3-3. Clustering

We applied clustering to better understand the patterns in the data and identify groups of steel plates with similar defect characteristics. Clustering helps in exploring the correlation between data without supervision.We clustered the following features.

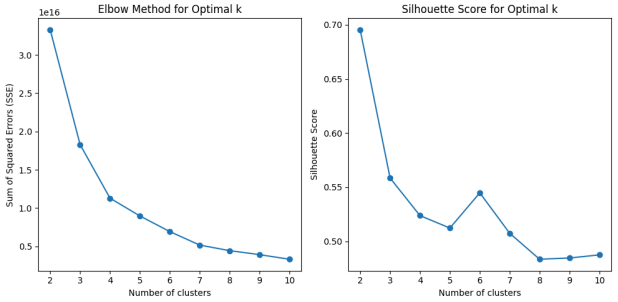
- X_Minimum
- Y_Minimum
- Pixels_Areas
- Sum_of_Luminosity
- Steel_Plate_Thickness

We employed the K-Means clustering algorithm. K-Means partitions the data into k clusters by minimizing the variance within each cluster. To determine the optimal number of clusters, we used the Elbow Method and the Silhouette Score:

Elbow Method: This method involves plotting the Sum of Squared Errors (SSE) against the number of clusters and looking for the "elbow point" where the SSE starts to decrease more slowly. The elbow point can indicate the optimal number of clusters without overfitting.

Silhouette Score: The silhouette score measures how similar a data point is to its own cluster compared to other clusters.Its range is -1 to 1 and a higher score can indicate better clusters. But higher scores do not guarantee the best. We calculated the silhouette score for different values of k to find the optimal clustering.

Based on Figure 3, we set the optimal number of clusters to 4.



[Figure 3: Number of clusters]

3-4. Log Scaling

Log scaling is a transformation technique that can help in dealing with skewed data distributions and reducing outliers. By applying a log function, we tried to make the data more normally distributed, which can help machine learning models to learn easily.

We applied log scaling to the following features.

- X_Perimeter
- Steel_Plate_Thickness

- Pixels_Areas

4. Modeling

We use ensemble technique to achieve the highest score. Also apply different weights for each defect, try to reflect the characteristics of each defect. Finally we use Stratified K-Fold considering the imbalance given targets.

4-1. Ensemble

Ensemble is a widely adopted technique to improve model performance in machine learning competition [7]. We can get each model's individual strengths and relieve their weaknesses. In this competition, we utilized an ensemble of the following models.

- XGBoost
- CatBoost
- LightGBM
- Histogram-Based Gradient Boosting

4-2. Individual Models and Hyperparameter Tuning

We provide a brief overview of each model and the hyperparameter tuning method.

XGBoost is known for its high efficiency and great classification performance with its ensemble method. It implements gradient boosting with additional features such as regularization to prevent overfitting [3].

CatBoost is designed to handle categorical features automatically and is particularly robust to overfitting [4]. This can be achieved by using ordered boosting, which prevents data leakage and constructs trees. It often requires less parameter tuning compared to other boosting algorithms [5].

LightGBM is a highly efficient gradient boosting framework that uses tree-based learning algorithms. It is well known for 'Exclusive Feature Bundling' technique which reduces dimension of data by extracting features that are not important. It is optimized for speed and performance, making it suitable for large datasets [6].

Histogram-Based Gradient Boosting is an implementation of gradient boosting that uses histograms to find the best splits, which can result in faster training times and reduced memory usage.

To optimize the performance of our four models before ensemble, we performed hyperparameter tuning using Optuna, a state-of-the-art optimization framework. The objective was to maximize the average AUC score. Using this framework, we can find the optimal hyperparameter fast and efficiently.

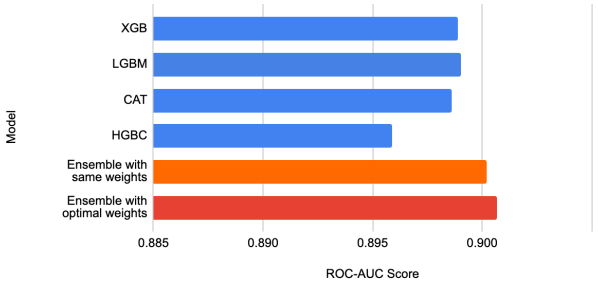
4-3. Different Weights for Each Target

In our competition, there are various types of defects, which require a different approach. We applied distinct weights to the four models for each defect type to reflect the characteristics of each defect type. We use the Scipy library to determine the optimal weights, and the resulting weights for each defect type are shown in Figure 4.

Class	XGB	CAT	LGBM	HGBC
Pastry	0.169	0.602	0.229	0.000
Z_Scratch	0.386	0.000	0.399	0.215
K_Scatch	0.326	0.079	0.448	0.147
Stains	0.281	0.263	0.304	0.152
Dirtiness	0.097	0.314	0.589	0.000
Bumps	0.340	0.328	0.288	0.045
Other_Faults	0.407	0.214	0.287	0.093

[Figure 4: Weights for each model]

None of the individual models achieved an ROC-AUC score higher than 0.9 in the cross-validation set. However the ensemble model with the same weights scored 0.90021. By fine-tuning the weights for each target, our ensemble model achieved a score of 0.90069, the highest score in our evaluations.

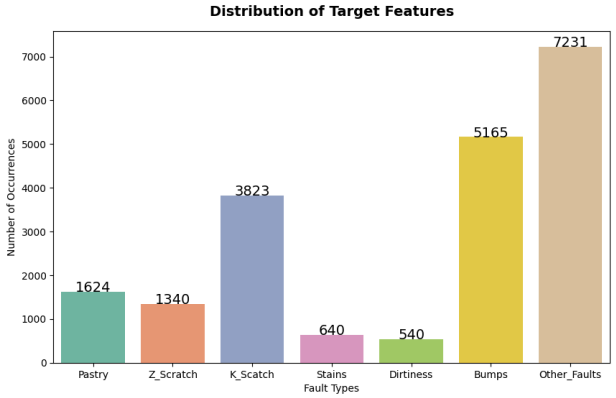


[Figure 5: ROC-AUC scores]

4-4. Cross-Validation Strategy

We used a Stratified K-Fold cross-validation to ensure our model's robustness. This method was important given the imbalance in our dataset, as illustrated by the distribution of target features in Figure 5. Stratified K-Fold cross-validation ensures that each fold has the same distribution unlike K-Fold.

By maintaining the proportional representation of each class in every fold, it provides a more reliable estimate of the model's performance across different segments of the data. This approach helps to prevent overfitting and generalize well to unseen data.



[Figure 5: Distribution of targets in train set]

5. Model Performance

By applying the techniques, we achieved 0.89652 in public score and 0.88905 in private score. Our private score dropped by approximately 0.075. However, our ranking moved from 78th to 74th. This suggests that we successfully mitigated the overfitting problem to some extent.

[Figure 6: Public score]

[Figure 7: Private score]

6. Conclusion

In this study, we developed a robust model for predicting defects in steel plates.

First, we performed exploratory data analysis (EDA), which included feature extraction, creating derived features, clustering, and log scaling.

Our model combines four GBDT algorithms, such as XGBoost, CatBoost, LightGBM, and Histogram-based Gradient Boosting Classification Tree (HGBC).

Finally, we applied different weights to each target to optimize the model for defect types. This ensemble model achieved an 0.88905 ROC-AUC score on the private leaderboard.

Using this model in steel manufacturing processes is expected to enhance quality control and improve operational efficiency. By accurately identifying and classifying defects, the model can reduce the incidence of defective products. Consequently, this can lead to reduced material waste, lower costs.

7. Future works

One of the challenges we faced was the lack of domain knowledge. In machine learning competitions, a deep understanding of the specific domain is crucial for effectively using the data. We hope we can build a more robust model by enhancing our knowledge of the given features.

Additionally, this competition only uses tabular dataset. For future work, we could expand the scope to include image data of steel plate defects. This would offer a more practical approach to addressing real-world problems.

References

[1] Reade, W., & Chow, A. (2024). Steel Plate Defect Prediction. Kaggle. https://kaggle.com/competitions/playground-series-s4e3

[2] Buscema, M., Terzi, S., & Tastle, W. (2010). Steel Plates Faults. UCI Machine Learning Repository. https://doi.org/10.24432/C5J88N

[3] Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794). ACM. https://doi.org/10.1145/2939672.2939785

[4] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: Unbiased boosting with categorical features. In Advances in Neural Information Processing Systems (Vol. 31, pp. 6638-6648). https://arxiv.org/abs/1706.09516

[5] Dorogush, A. V., Ershov, V., & Gulin, A. (2018). CatBoost: Gradient boosting with categorical features support. arXiv preprint arXiv:1810.11363. https://arxiv.org/abs/1810.11363

[6] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. (2017). LightGBM: A highly efficient gradient boosting decision tree. In Advances in Neural Information Processing Systems (Vol. 30, pp. 3146-3154). https://arxiv.org/abs/1712.01860

[7] Ren, Y., Zhang, L., & Suganthan, P. N. (2016). Ensemble Classification and Regression-Recent Developments, Applications and Future Directions [Review Article]. IEEE Computational Intelligence Magazine, 11(1), 41-53. https://doi.org/10.1109/MCI.2015.2471235

[8] 임응순. (2010). 국내 철강산업의 경제적 파급효과. POSRI 경영경제연구, 10(2), 5-27.