|  |  |  |
| --- | --- | --- |
| Nama Akun Kaggle | : | Whisnumurty GA |
| Link Akun Kaggle | : | <https://www.kaggle.com/whisnumurtyga> |
| Nama Competition | : | Steel Plate Defect Prediction |
| Deskripsi Challenge | : | Mengembangkan model prediktif yang dapat mengidentifikasi dan memprediksi keberadaan cacat pada pelat baja. Peserta diberikan kumpulan data yang berisi informasi tentang berbagai atribut fisik dan kimia dari pelat baja, serta label yang menunjukkan apakah pelat baja tersebut mengandung cacat atau tidak. |

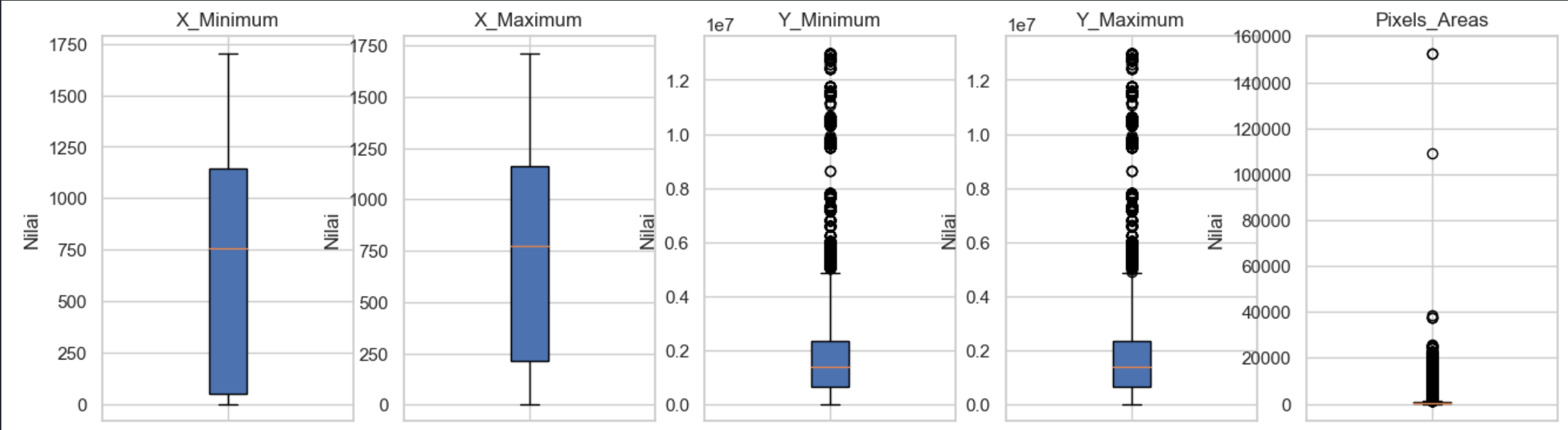
# Step 1 – Data Understanding

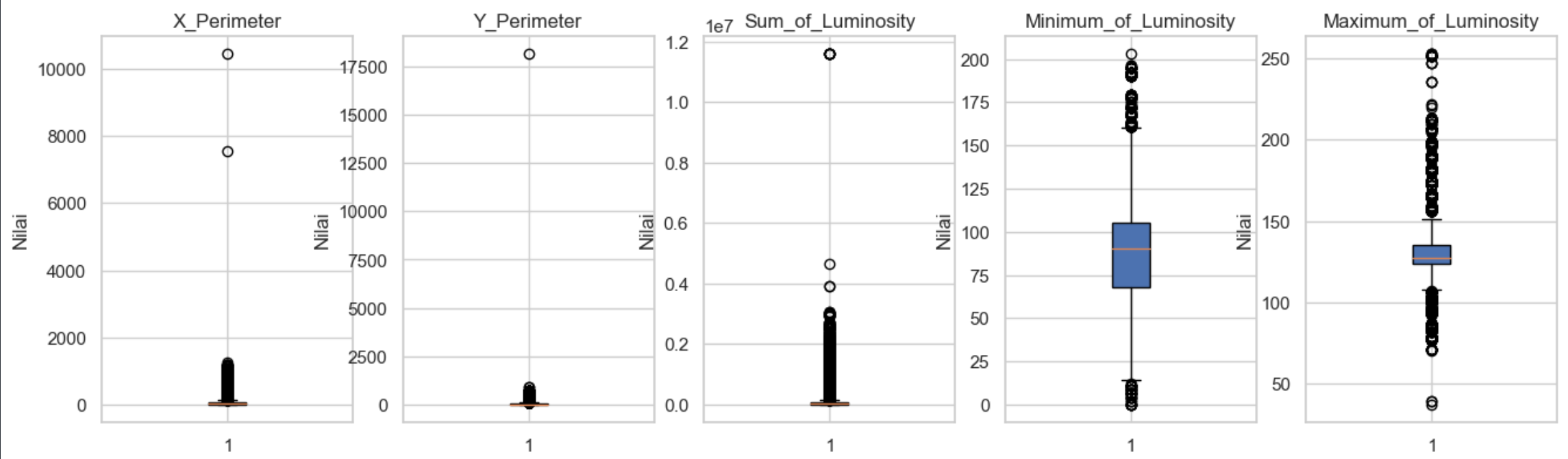
|  |  |
| --- | --- |
| Nama Feature | Deskripsi |
| Id | ID unik untuk setiap sampel data. |
| X\_Minimum, X\_Maximum, Y\_Minimum, Y\_Maximum | Koordinat batas persegi panjang yang mengelilingi cacat pada pelat baja. |
| Pixel\_Areas | Jumlah piksel yang tercakup oleh cacat tersebut. |
| X\_Perimeter, Y\_Perimeter | Panjang perimeter (keliling) dalam arah X dan Y dari cacat. |
| Sum\_of\_Luminosity, Minimum\_of\_Luminosity, Maximum\_of\_Luminosity | Informasi tentang luminositas cacat, termasuk jumlah total luminositas, luminositas minimum, dan maksimum. |
| Length\_of\_Conveyer | Panjang konveyor di mana pelat baja bergerak. |
| TypeOfSteel\_A300, TypeOfSteel\_A400 | Variabel biner yang menunjukkan jenis baja (A300 atau A400). |
| Steel\_Plate\_Thickness | Ketebalan pelat baja. |
| Edges\_Index, Empty\_Index, Square\_Index, Outside\_X\_Index, Edges\_X\_Index, Edges\_Y\_Index, Outside\_Global\_Index | Berbagai indeks yang mungkin dihitung dari citra pelat baja. |
| LogOfAreas, Log\_X\_Index, Log\_Y\_Index, Orientation\_Index, Luminosity\_Index, SigmoidOfAreas | Fitur-fitur lain yang dihitung dari citra pelat baja. |
| Pastry, Z\_Scratch, K\_Scatch, Stains, Dirtiness, Bumps, Other\_Faults: | Label yang menunjukkan jenis cacat yang ada pada pelat baja. Misalnya, "Pastry" mungkin menunjukkan adanya cacat berbentuk pastry, dan seterusnya. |

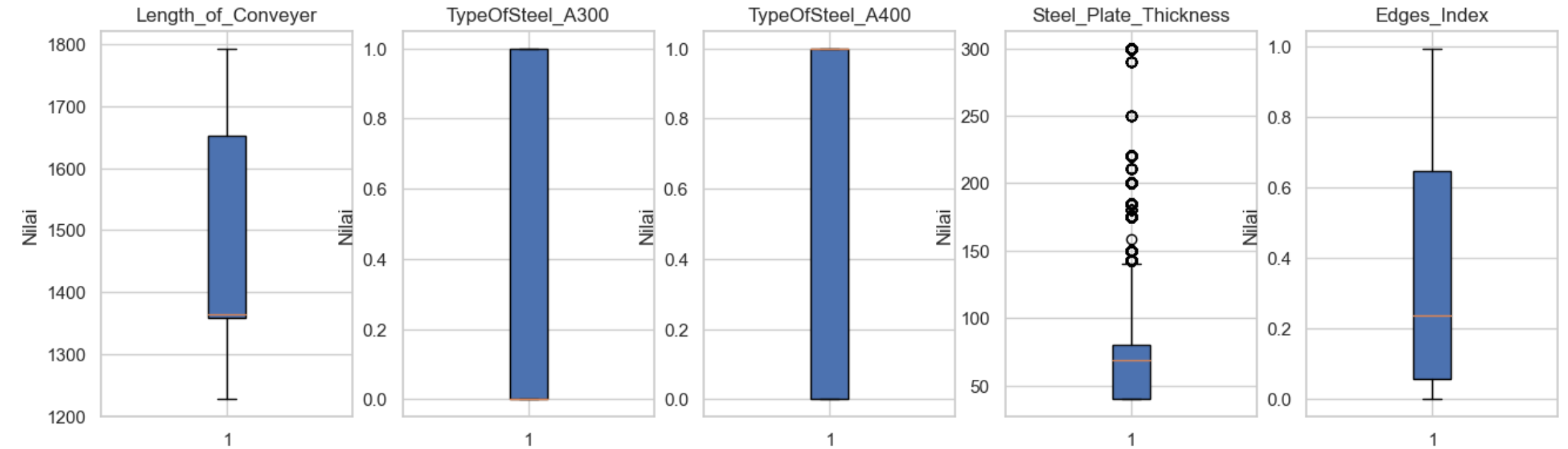
# Step 2 – Set Requirements

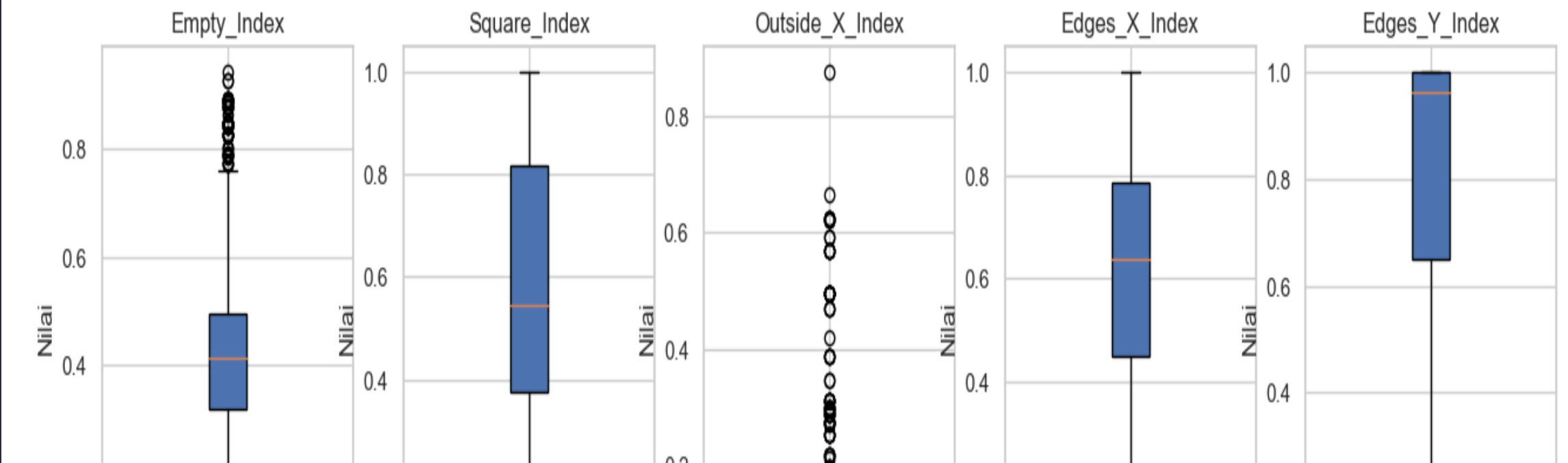
|  |  |
| --- | --- |
| Library | Fungsi |
| NumPy | melakukan operasi numerik untuk bekerja dengan array dan matriks |
| Pandas | Manipulasi dan analisis data |
| Pretty Table | Membuat table rapi dan mudah dibaca |
| Matplotlib & Seaborn | Visualisasi Data |
| DeepCopy | Mencopy model |
| Partial | Untuk membuat fungsi baru dengan beberapa argumen dari fungsi yang ada. |
| gc | Garbage Collector untuk memerikan control atas pemulihan memori |
| Scikit Learn | Untuk machine learning dan analisis data. Serta menyediakan berbagai algoritma machine learning, evaluasi model, dan alat-alat pra-pemrosesan data. |
| Optuna | Mengoptimasi parameter secara otomatis |
| XGBoost | Library untuk pemodelan prediktif decision tree based sebagai model dasar |
| CatBoost | Library untuk menangani data kategorikal |

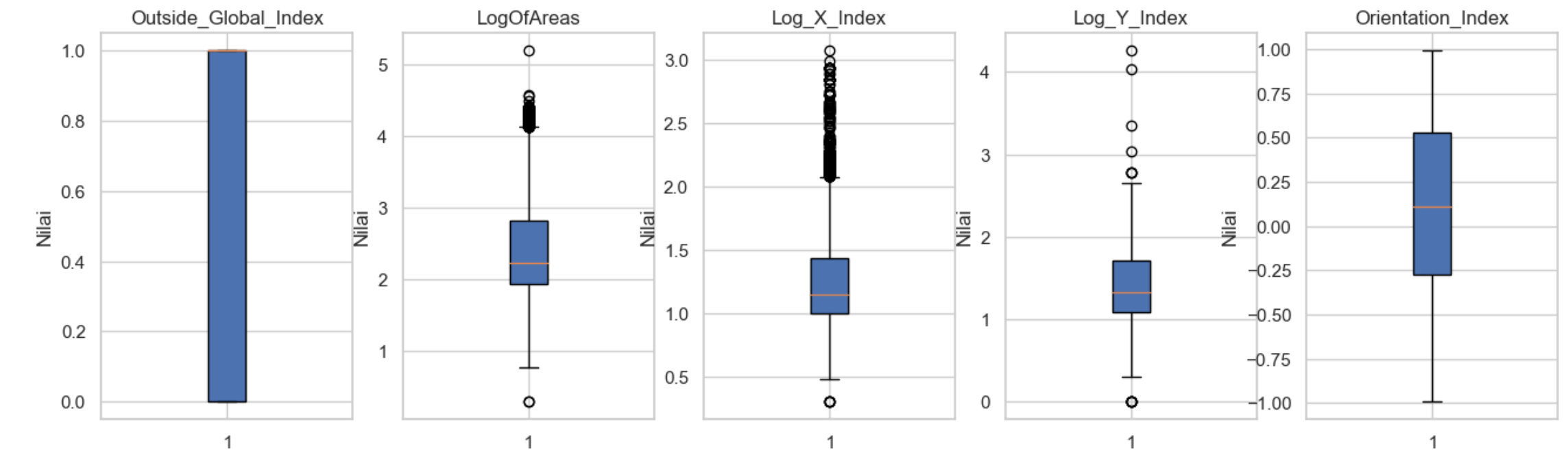
# Step 3 – Exploratory Data Analysis

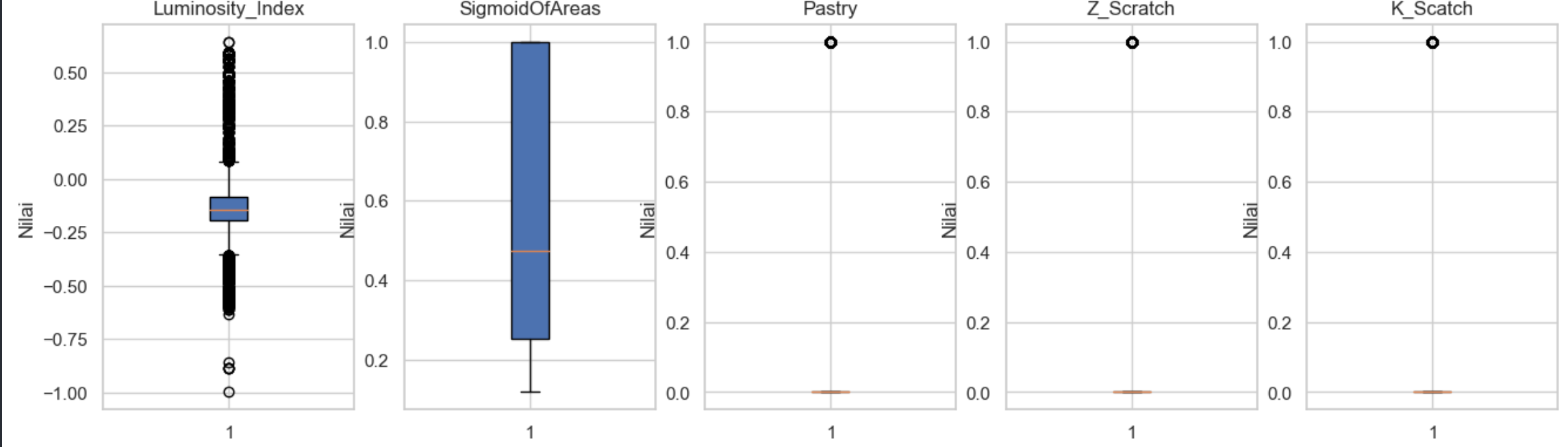


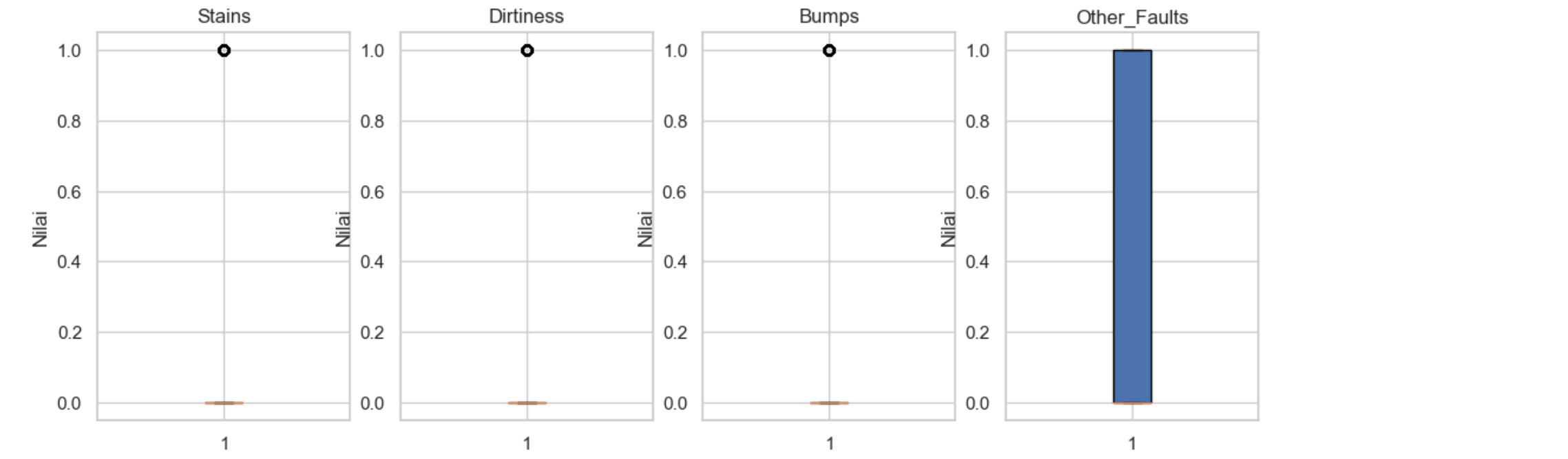




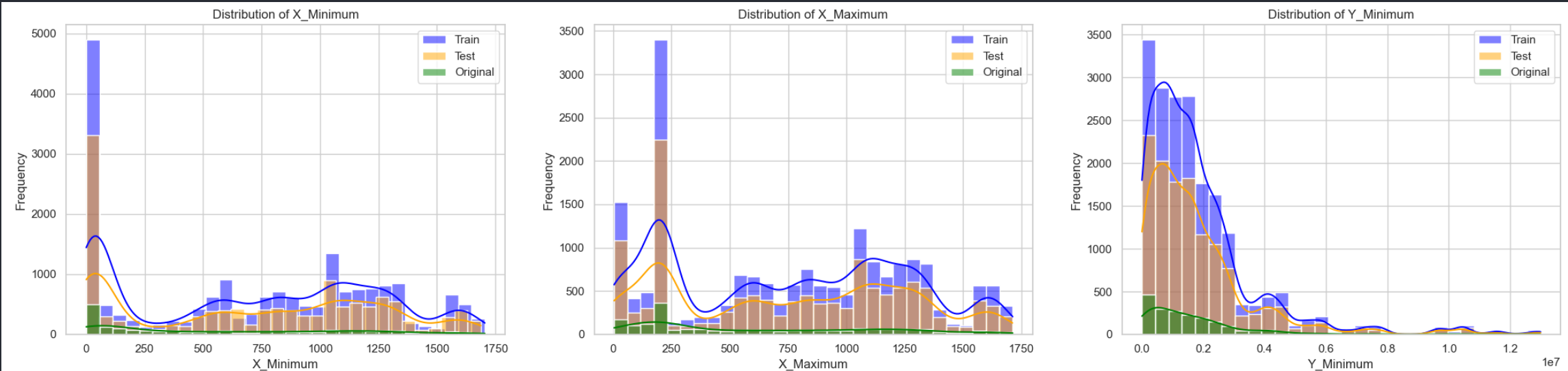


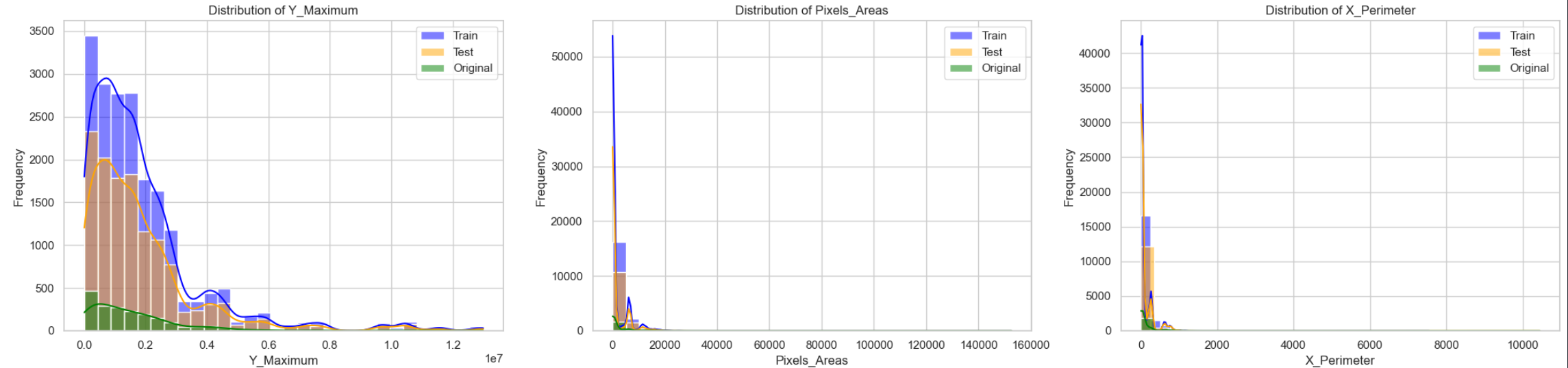






Dari hasil boxplot tersebut dapat kita ketahui bahwa terdapat kolom kategorik yang sudah dijadikan bentuk numerik seperti (Pastry, Z\_Scratch, K\_Scatch, Strains, Dirtiness, Bumps, Other Faults, Outside\_Global Index), dari box plot tersebut kita juga dapat melihat beberapa feature terutama pada y\_minimum dan y\_maximum (artinya Panjang baja secara vertical sangat bervariasi)

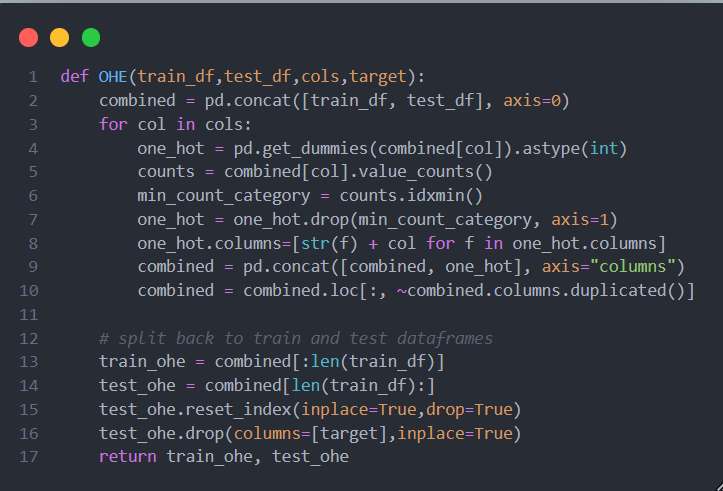




Dari grafik histogram tersebut bis akita lihat bahwa distribusi data numerik memiliki pola kurva distribusi yang agak berbeda dari data original. Hal tersebut menandakan bahwa data sintesis yang dihasilkan dari data original memiliki pola yang berbeda.

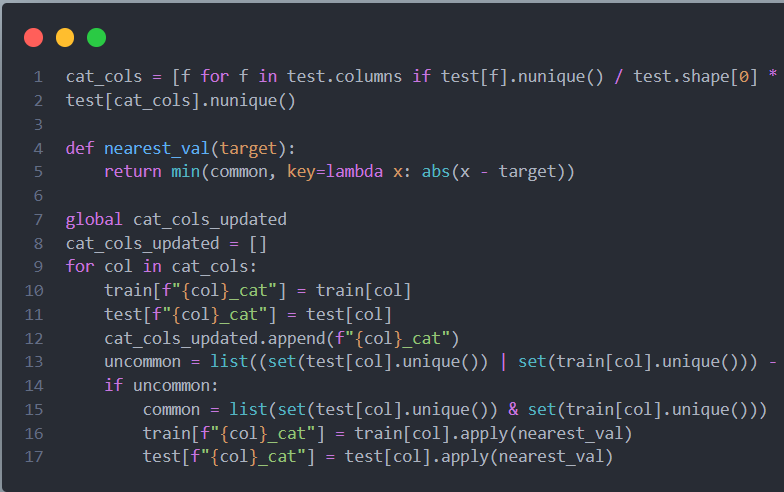
# Step 4 - Feature Engineering

## 4.1 One Hot Encoding



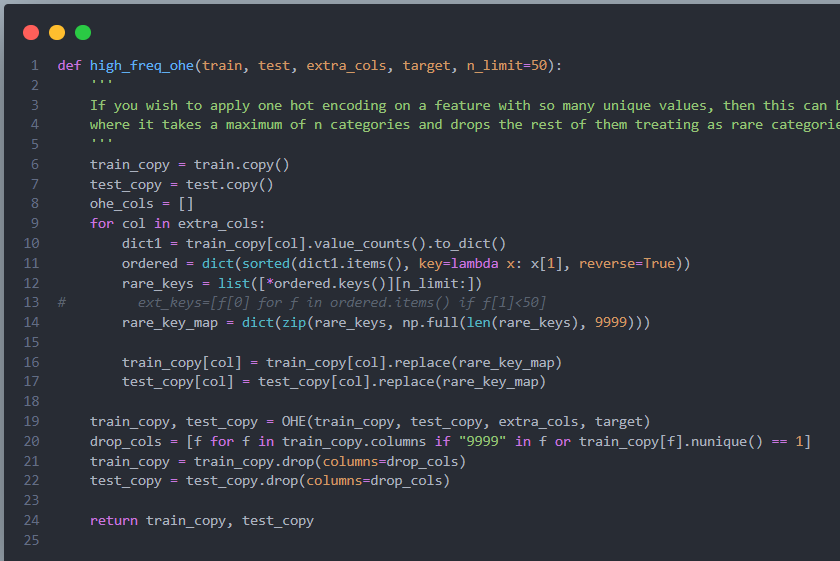
Melakukan proses one-hot encoding (OHE) pada kolom-kolom tertentu dari data latih (train\_df) dan data uji (test\_df), dengan menghapus kategori yang memiliki frekuensi terendah. Proses ini bertujuan untuk mengubah variabel kategorikal menjadi representasi biner yang dapat digunakan oleh model pembelajaran mesin untuk meningkatkan kinerja dalam melakukan prediksi.

## 4.2 Stabilize Categorical Columns



Mengelompokkan variabel kategorikal yang memiliki jumlah kategori yang sedikit dan jarang muncul dalam data menjadi kategori baru yang lebih umum, berdasarkan nilai terdekat dari kategori yang umum di setiap data latih dan data uji. Hal ini dilakukan dengan mengecek variabel kategorikal dalam data uji, kemudian mencari nilai unik yang kurang dari 5% dari total baris dan lebih dari 2 nilai unik. Setelah itu, dilakukan proses pembuatan kategori baru dengan memilih nilai terdekat dari nilai-nilai umum yang terdapat dalam data latih dan data uji untuk setiap variabel kategorikal yang diproses. Variabel-variabel kategorikal yang telah diperbarui kemudian disimpan dalam list cat\_cols\_updated untuk penggunaan selanjutnya.

## 4.3 High Freq OHE



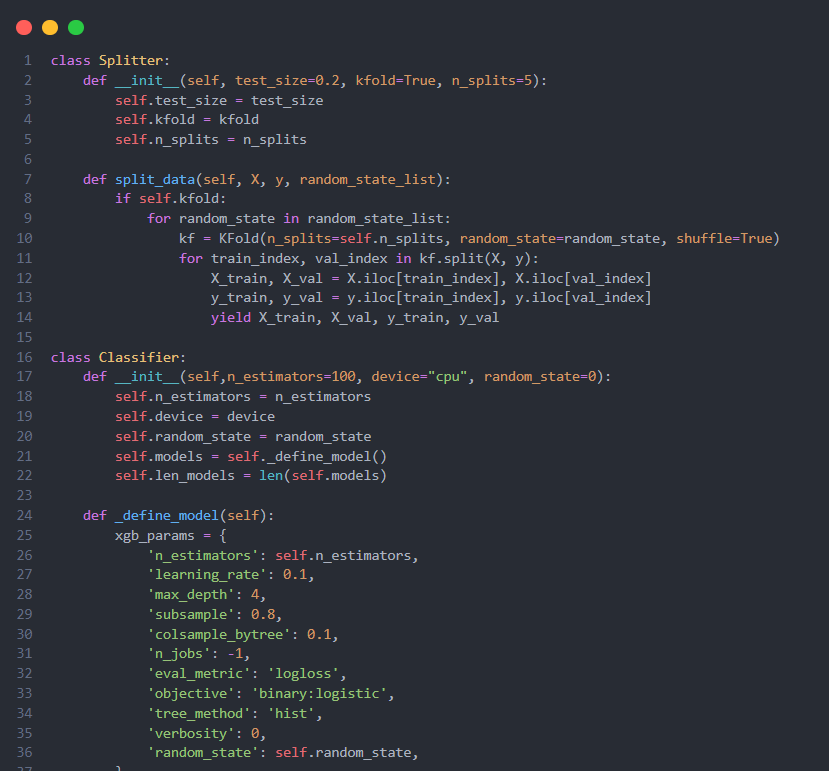
Menerapkan one-hot encoding pada fitur dengan jumlah nilai unik yang banyak dengan batasan tertentu. Ketika fitur memiliki jumlah kategori yang melebihi batas yang ditentukan (n\_limit), fitur akan dianggap memiliki kategori langka. Fitur-fitur dengan kategori langka ini akan diubah menjadi nilai yang disesuaikan (misalnya, 9999) dalam kedua dataset train dan test. Setelah itu, fungsi OHE dipanggil untuk melakukan one-hot encoding pada fitur-fitur yang telah dimodifikasi. Kemudian, kolom-kolom yang memiliki nilai 9999 atau memiliki satu nilai unik saja dihapus dari kedua dataset untuk mengurangi dimensi data yang dihasilkan. Setelah itu, dataset train dan test yang telah dimodifikasi kembali dikembalikan sebagai output dari fungsi ini.

## 4.4 Category Encoding

M­­elakukan encoding pada variabel kategorikal dalam dataset train dan test. Pertama-tama, fungsi ini membuat salinan dari dataset train dan test. Kemudian, untuk setiap variabel kategorikal yang telah diperbarui (cat\_cols\_updated), fungsi ini menghitung frekuensi masing-masing nilai, membuat kolom baru yang berisi jumlah kemunculan setiap nilai kategorikal (feature + "\_count") serta memberikan peringkat pada setiap nilai kategorikal berdasarkan jumlah kemunculannya (feature + "\_count\_label"). Jika jumlah nilai unik dari variabel kategorikal kurang dari atau sama dengan 5, variabel tersebut akan dianggap kategorikal dan akan diterapkan one-hot encoding. Jika tidak, variabel tersebut akan diubah menggunakan fungsi high\_freq\_ohe. Selanjutnya, model Machine Learning akan dilatih menggunakan fitur-fitur yang telah dihasilkan untuk memprediksi target, dan diukur kinerjanya menggunakan AUC (Area Under the ROC Curve). Fitur dengan kinerja terbaik akan dipilih dan kolom-kolom yang berkorelasi kuat dengan fitur terbaik akan dihapus untuk menghindari multikolinearitas. Informasi mengenai fitur terbaik beserta kinerjanya akan ditambahkan ke dalam sebuah tabel yang ditampilkan, dan dataset train dan test yang telah dimodifikasi akan dikembalikan sebagai output dari fungsi ini.

# Step 5 – Modelling

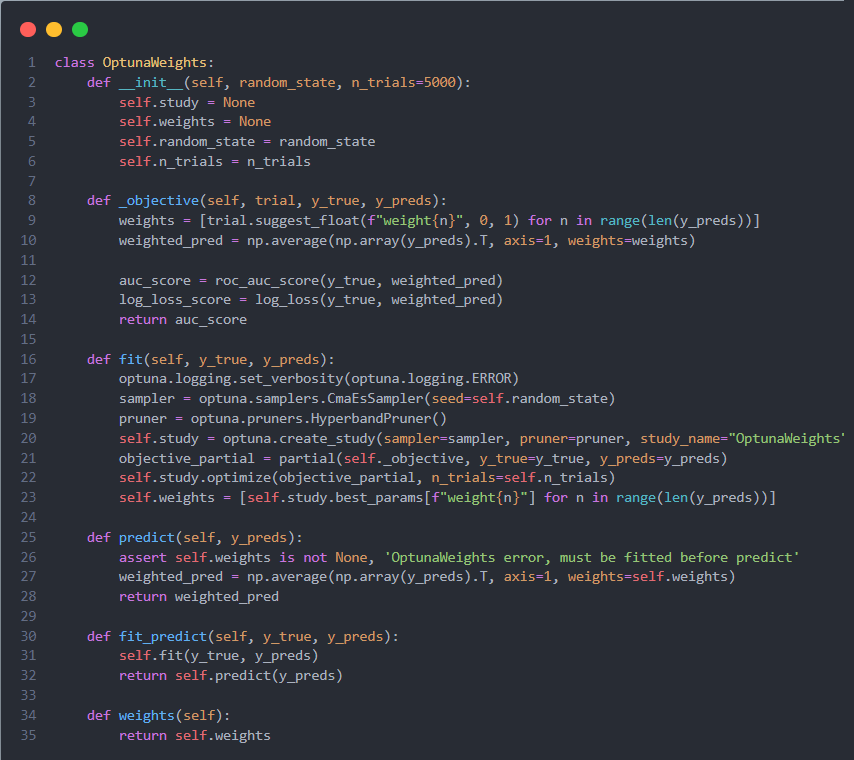
## 5.1 Create Model



Kelas `Splitter` bertujuan untuk memisahkan data menjadi set pelatihan dan validasi menggunakan teknik pengacakan K-Fold Cross Validation. Pengguna dapat menentukan ukuran data uji (`test\_size`) dan jumlah lipatan (`n\_splits`) untuk validasi silang. Metode `split\_data()` digunakan untuk membagi data menjadi set pelatihan dan validasi.

Kelas `Classifier` bertujuan untuk mendefinisikan beberapa model klasifikasi, yaitu XGBoost (`xgb`) dan CatBoost (`cat` dan `cat\_sym`). Pengguna dapat menentukan jumlah estimator (`n\_estimators`) dan perangkat (`device`) yang ingin digunakan. Pada bagian ini, parameter-parameter untuk setiap model didefinisikan, seperti jumlah estimator, kedalaman maksimum, tingkat pembelajaran, dan lain-lain. Model-model ini disimpan dalam atribut `models` dari objek `Classifier`.

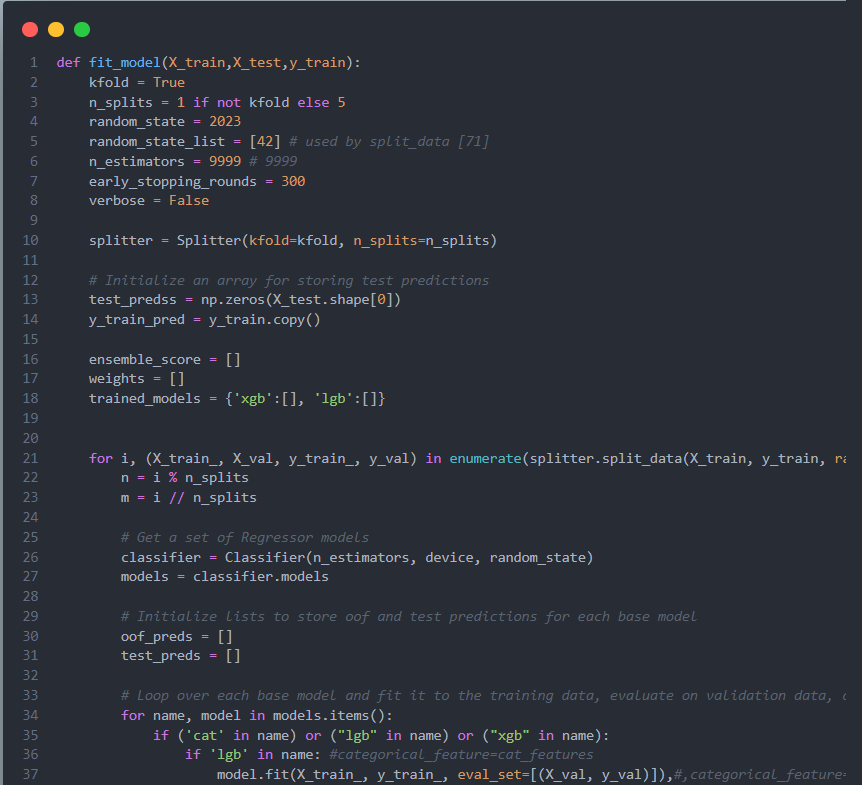
## 5.2 Optimize Ensemble Weight



Kelas OptunaWeights bertujuan untuk mencari bobot terbaik yang diberikan kepada prediksi dari beberapa model. Ini berguna ketika Anda memiliki beberapa model yang menghasilkan prediksi untuk tugas klasifikasi atau regresi, dan Anda ingin menggabungkan prediksi-prediksi ini menjadi satu prediksi tunggal dengan bobot yang dioptimalkan. Kelas ini menggunakan Optuna, sebuah library untuk optimasi hiperparameter, untuk mencari bobot terbaik yang memberikan hasil optimal berdasarkan metrik evaluasi yang dipilih (dalam hal ini, Area Under the ROC Curve atau AUC).

* Metode fit() digunakan untuk melakukan pencarian bobot terbaik berdasarkan prediksi dari beberapa model.
* Metode predict() digunakan untuk menghasilkan prediksi tunggal menggunakan bobot terbaik yang telah ditemukan.
* Metode fit\_predict() adalah gabungan dari fit() dan predict(), yang digunakan untuk melakukan pencarian bobot terbaik dan menghasilkan prediksi tunggal dalam satu langkah.
* Metode weights() digunakan untuk mengembalikan bobot yang telah ditemukan setelah proses pelatihan.

## 5.3 Fit the Model



Fungsi fit\_model bertujuan untuk melatih model ensemble menggunakan beberapa model dasar seperti XGBoost dan CatBoost, serta mengevaluasi kinerjanya menggunakan metrik ROC AUC. Berikut adalah penjelasan singkat tentang bagaimana fungsi ini bekerja:

* Fungsi ini membagi data latih menjadi set pelatihan dan validasi menggunakan teknik K-Fold Cross Validation jika kfold diatur sebagai True, dan menggunakan satu lipatan jika tidak. Ini dilakukan menggunakan objek Splitter.
* Untuk setiap lipatan, fungsi akan melatih setiap model dasar (XGBoost, CatBoost, dll.) menggunakan data pelatihan dan mengevaluasinya menggunakan data validasi.
* Setelah model-model dasar dilatih, fungsi akan menggunakan Optuna untuk menemukan bobot terbaik yang akan diberikan kepada prediksi dari model-model tersebut, sehingga menghasilkan prediksi ensemble yang optimal.
* Fungsi akan menggabungkan prediksi dari model-model dasar dengan bobot yang dioptimalkan untuk menghasilkan prediksi ensemble akhir.
* Kinerja ensemble diukur menggunakan metrik ROC AUC dan dicetak.
* Bobot rata-rata dan standar deviasi bobot dari setiap model dasar dicetak.
* Prediksi ensemble yang dihasilkan dikembalikan sebagai output dari fungsi ini.

# Step 6 – Submission



Dalam percobaan ini, dilakukan iterasi untuk setiap kolom target dalam dataset. Setiap iterasi dimulai dengan memodifikasi dataset latih dan uji menggunakan fungsi cat\_encoding, yang mengkodekan variabel kategorikal, dan menghasilkan set fitur final yang akan digunakan untuk pelatihan model. Selanjutnya, fitur-fitur ini dinormalisasi menggunakan StandardScaler untuk memastikan distribusi yang seragam, penting untuk beberapa jenis model seperti SVM. Proses pelatihan model kemudian dilakukan menggunakan fungsi fit\_model, yang melibatkan pemilihan model ensemble dan pencarian bobot terbaik untuk prediksi yang dioptimalkan. Prediksi akhir dihasilkan untuk setiap kolom target dalam dataset uji, dan hasilnya disimpan dalam dataframe submission. Proses ini diulang untuk setiap kolom target, dan setiap langkah dalam iterasi diberi nomor untuk memantau kemajuan. Ini menyajikan suatu pendekatan sistematis dalam menghasilkan prediksi yang akurat untuk setiap kolom target dalam dataset.

# Step 7 – Generalization Ensemble

Kode ini digunakan untuk menggabungkan hasil dari beberapa file submission dengan memberikan bobot tertentu pada masing-masingnya. Pertama, dua file submission (sub1 dan submission) dibaca. Selanjutnya, bobot diberikan pada setiap file submission dan file-file submission ini digabungkan menjadi satu dalam sebuah list. Fungsi ensemble\_mean kemudian digunakan untuk menghitung ensemble dari file submission dalam list tersebut, dengan mempertimbangkan tiga jenis rata-rata: Arithmetic Mean (AM), Geometric Mean (GM), dan Harmonic Mean (HM). Dalam kasus ini, digunakan AM. Hasil ensemble ini kemudian disimpan dalam sebuah file CSV baru dengan nama submission-ensemble.csv untuk digunakan lebih lanjut. Ini memberikan cara efisien untuk menggabungkan hasil prediksi dari berbagai model atau strategi yang telah dikembangkan.

# Result

Akurasi Public : 0.89598

Akurasi Private : 0.88877

