1) Goal

We want to predict the “efficiency” value accurately so that our Kaggle leaderboard score becomes lower (better).

2) Problem Type

This is a regression problem, meaning we predict a continuous number, not a class.

3) Why Use Cross-Validation (5-Fold)

Instead of training one model and hoping it generalizes, we:

- Split the training data into 5 parts
- Train on 4 parts, validate on the remaining 1
- Repeat for all 5 combinations

This gives a more reliable score and avoids overfitting.

4) Why Use Target Encoding Instead of One-Hot

- One-hot encoding makes many columns when there are many categories.
- Target Encoding replaces each category with the average target (efficiency) for that category.
- This reduces overfitting and improves leaderboard performance.

5) Why Optional Log1p Transformation

If efficiency values are positive:

- We convert y to log1p(y)
- Model learns more smoothly
- After prediction we apply expm1() to get back original values.

6) Why XGBoost

XGBoost is powerful for tabular data because it:

- Handles nonlinear relationships
- Regularizes to reduce overfitting
- Supports early stopping to avoid training too long

7) Why CatBoost (Optional)

CatBoost can directly handle categorical features without encoding.

If installed, we blend it with XGBoost because:

- XGBoost + CatBoost together perform better than either alone.

8) Model Blending

We find a best blending weight α so:

final_prediction = α * XGBoost_pred + (1 - α) * CatBoost_pred

This improves accuracy.

9) Clipping Predictions

We make sure predictions stay within the realistic range of the target values:

prediction = min(max(prediction, y_min), y_max)

10) Final Step: Submission

We create submission.csv by replacing the efficiency column in sample_submission.csv with our predictions.

We upload submission.csv to Kaggle.