Code Quality Classification

Imagine you are a lead data scientist in charge of a critical project within your organization. The project involves developing a complex software system that is integral to the success of a new product launch. As the project progresses, there is a growing concern about the overall quality of the software codebase, and management has decided to conduct a thorough software quality assessment. The development team has been diligently working on project, and the codebase has evolved over time. However, as the project is reaching a crucial milestone, there is a need to ensure that the software meets the highest quality standards to minimize the risk of defects and ensure long-term maintainability. To initiate the software quality assessment, the team has gathered a dataset containing various metrics related to the software's complexity, size, structure, and potential defects. Your task as the lead data scientist is to leverage this dataset to perform a comprehensive software quality assessment. The goal is to analyze the provided metrics and identify key insights regarding the overall quality of the software. Your findings will be crucial in making informed decisions about areas that need improvement, potential risks, and the overall health of the software codebase.

Train data: 74794

Test data: 32056

Dataset Description:

- id: Unique identifier for each record.
- McCabeLineCount:Total number of lines in the software as measured by McCabe.
- McCabeCyclomaticComplexity: A metric indicating the number of linearly independent paths through a program's source code.
- McCabeEssentialComplexity: Measure of the essential control flow complexity of the software.
- McCabeDesignComplexity: Design-level complexity based on McCabe metrics.
- HalsteadTotalOperatorsOperands:Total number of distinct operators and operands in the software.
- HalsteadVolume: Halstead's software science metric representing the program's size.
- HalsteadProgramLength: Length of the program as measured by Halstead.
- HalsteadDifficulty: Difficulty level of understanding the software based on Halstead metrics.
- HalsteadIntelligence: Intelligence level required to understand the software.
- HalsteadEffort:  Effort required to develop and maintain the software.
- HalsteadB: A parameter in the Halstead metrics.
- HalsteadTimeEstimator: Estimated time required to develop the software.
- HalsteadLineCountCode: Number of lines of code in the software.
- HalsteadLineCountComment: Number of lines of comments in the software.
- HalsteadLineCountBlank: Number of blank lines in the software.
- HalsteadLineCountCodeAndComment: Number of lines containing both code and comments.
- UniqueOperators: Number of unique operators used in the software.
- UniqueOperands: Number of unique operands used in the software.
- TotalOperators: Total number of operators used in the software.
- TotalOperands: Total number of operands used in the software.
- BranchCount: Number of branches in the software code.
- defects: Indicator variable for the presence of defects in the software.
- CodeDensity: Density of code in the software.
- OperatorToOperandRatio: Ratio of operators to operands in the software.
- CommentDensity: Density of comments in the software.

- ComplexityEfficiency: Efficiency of the software in managing complexity.
- OperandsPerOperator: Average number of operands per operator.
- CodeAndCommentRatio: Ratio of code lines to comment lines in the software.
- CodeAge: Age of the codebase.
- CodeLanguage: Programming language used for development.
- CodeSizeCategory: Categorization of code size.
- CodeType: Type or category of the software code.
- CodeQuality: Overall quality rating of the software code.
- QualityScore: A composite score representing the overall quality of the software.
- IsDeprecated: Indicator variable for code deprecation.
- TeamSize: Size of the development team.