Automatic Programming Problem Difficulty Prediction

A Machine Learning and NLP Based Approach

1. Introduction

Online coding platforms host thousands of programming problems with varying difficulty levels. Accurately labeling these problems is essential for learners, instructors, and competitive programmers. This project proposes an automated system that predicts the difficulty level of programming problems using machine learning and natural language processing techniques.

2. Problem Statement

Manual difficulty labeling is subjective and inconsistent across platforms. The goal of this project is to build a system that can automatically analyze a problem statement and predict: A categorical difficulty label (Easy, Medium, Hard) A continuous numerical difficulty score

3. Dataset Description

A custom dataset of 4,112 programming problems was collected in JSON Lines (.jsonl) format. Each entry represents a single programming problem and includes the title, description, input/output formats, sample test cases, difficulty class, and difficulty score.

4. Data Preprocessing

All textual fields were merged into a single description to preserve context. The text was converted to lowercase and missing values were handled using empty string substitution. This ensured consistency and robustness during feature extraction.

5. Feature Engineering

A hybrid feature extraction strategy was used: **Character-level TF-IDF:** Captures symbols, formatting, and structural patterns. **Keyword-based features:** Detects algorithms and data structures. **Numeric features:** Includes text length, numeric counts, constraints, and complexity terms. Numeric features were standardized using StandardScaler.

6. Model Architecture

Two XGBoost models were trained using the same feature representation: **XGBoost Classifier:** Predicts Easy / Medium / Hard. **XGBoost Regressor:** Predicts a continuous difficulty score.

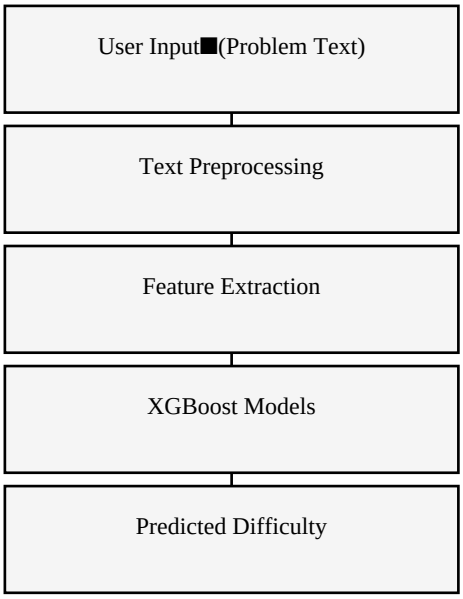
7. Evaluation Metrics

Classification performance was evaluated using accuracy, precision, recall, and F1-score, achieving approximately 52% accuracy. Regression performance was measured using RMSE (1.99), MAE (1.64), and R^2 score (0.17).

8. Web Application

A Streamlit-based web interface allows users to input problem descriptions and receive real-time predictions. The application loads pre-trained models and preprocessors using joblib.

9. System Architecture



10. Conclusion & Future Work

The proposed system demonstrates that problem difficulty can be inferred from textual descriptions. Future improvements include transformer-based models, larger datasets, and platform-specific tuning.