ML Project Report: Signal Cluster Classification Dataset

Team: Teen Bhai Teeno Tabahi

- Yash Gupta (IMT2023125)
- Pranay Kelotra (IMT2023563)
- Hitanshu Seth (IMT2023100)

Github Link: https://github.com/Hitanshu078/ML-Checkpoint-2/tree/main/Signal_cluster_classification

1. Task

The objective of this project is to develop a multiclass classification model capable of predicting the **personality cluster** (`category`) of an observation based on its signal measurements.

- **Goal:** Predict the categorical target variable `category` (Group_A, Group_B, or Group_C) using the numerical features provided.
- **Problem Type:** This is a **Multiclass Classification** task.
- **Evaluation Metric:** The models are evaluated using the **Macro F1 Score**. This metric is chosen to ensure that all clusters are treated with equal importance, preventing the model from being biased toward the most frequent class (Group_B).

2. Dataset and Features Description

- The dataset consists of **1,444** synthetic records used for training. Each record represents a sample with two signal-based predictor variables and one target label.
- **Features (Predictors):**
  - `signal_strength` (Numerical): A continuous variable representing the intensity of the signal.
  - `response_level` (Numerical): A continuous variable representing the magnitude of the response.
- **Target Variable:**
  - `category` (Categorical): The class label for the observation. It contains three distinct groups:
    - **Group_B:** The majority class (approx. 49% of data).
    - **Group_C:** The second most frequent class (approx. 33% of data).
    - **Group_A:** The minority class (approx. 18% of data).

3. Exploratory Data Analysis and Preprocessing

3.1. Initial Data Inspection

The dataset was loaded and examined for structure and integrity.

- **Shape:** The training set contains 1,444 rows and 4 columns (`sample_id`, `signal_strength`, `response_level`, `category`).
- **Data Types:** The features are floating-point numbers (`float64`), and the target is an object/string.

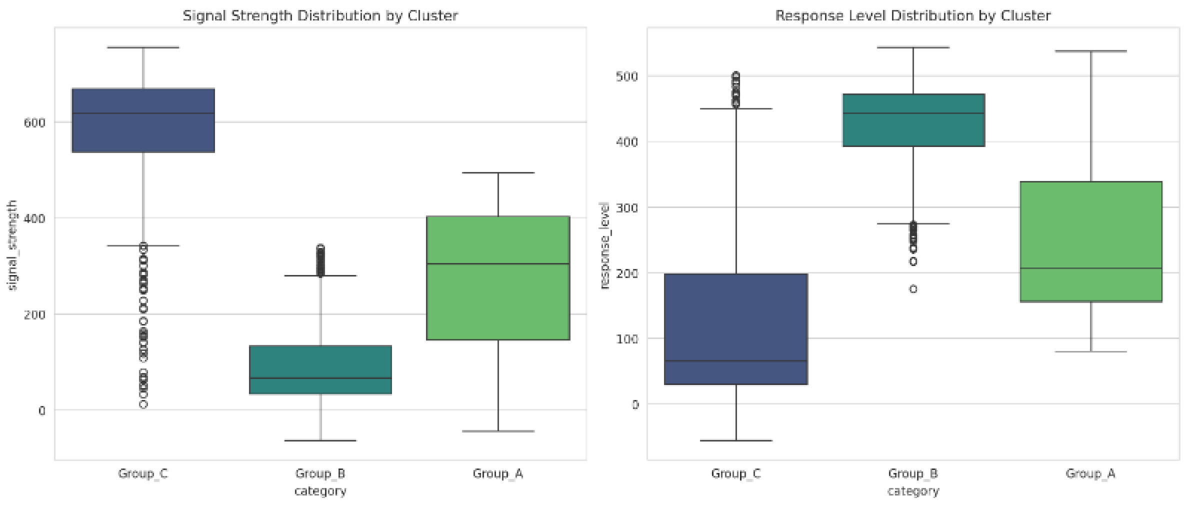
3.2. Handling Missing and Duplicate Values

- **Missing Values:** The dataset was checked for null values. No missing data was found in any column.
- **Duplicate Values:** The dataset was checked for duplicate records based on the feature columns. No duplicate entries were found.

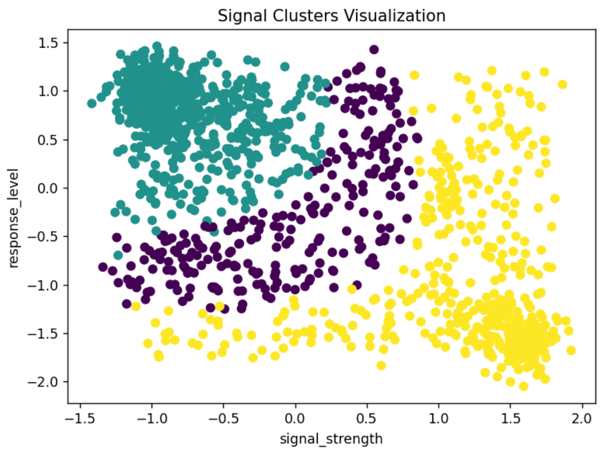
3.3. Exploratory Data Analysis (EDA)

Visual analysis was performed to understand the spatial distribution of the clusters .

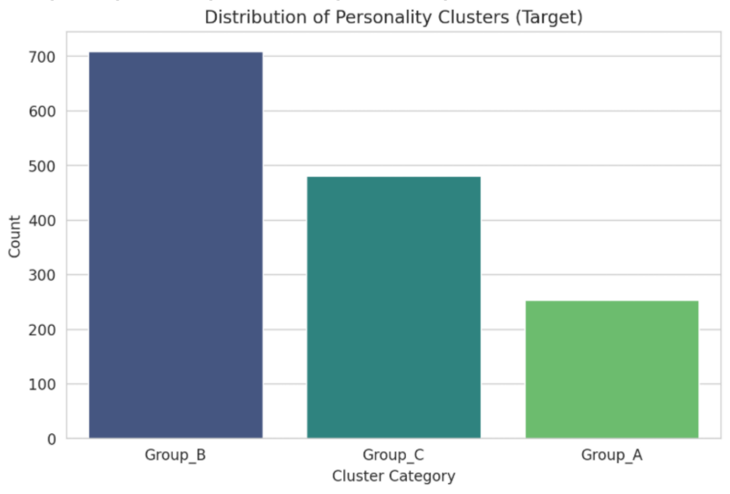
- **Cluster Separation:**
    - **Group_B** tends to have **low signal strength** and **high response levels** (Top-Left region).
    - **Group_C** tends to have **high signal strength** and **low response levels** (Bottom-Right region).
    - **Group_A** is located in the intermediate region (Central), showing significant overlap with the boundaries of the other two groups.



- **Non-Linearity:** The scatter plot reveals that the decision boundaries between groups (especially separating Group_A from the others) are not perfectly vertical or horizontal lines. This suggests that non-linear models (like k-NN or SVM) may outperform simple linear models.



- **Class Imbalance:** There is a notable imbalance, with Group_B having nearly three times as many samples as Group_A. This confirms the necessity of using the **Macro F1 Score** and potentially using classweighting strategies during training.



3.4. Data Preprocessing

The following steps were taken to prepare the data for modeling:

- **Dropping ID:** The `sample_id` column was removed as it is an identifier and holds no predictive power.
- **Target Encoding:** The categorical target `category` was encoded into numerical values (e.g., Group_A=0, Group_B=1, Group_C=2) for model compatibility.
- **Feature Scaling:** Both `signal_strength` and `response_level` were standardized using **StandardScaler**.
  - *Reasoning:* Since algorithms like k-Nearest Neighbors (k-NN) and Support Vector Machines (SVM) calculate Euclidean distances, it is crucial that both features are on the same scale (mean=0, std=1) to prevent one feature from dominating the distance metric.

3.5. Feature Engineering

To assist the model in distinguishing the complex, non-linear boundaries between clusters (especially the central "Group A"), three specific features were engineered:

- **Interaction:** (`signal * response`) Captures the combined effect of the two signals, helping to distinguish regions where both are high or low.

- **Magnitude:** (`sqrt(signal^2 + response^2)`) Represents the geometric distance from the origin. This is a crucial feature because Group A appears centrally located, while Groups B and C are further out; "magnitude" effectively acts as a radius feature.

Models Used for Training

- Given the potential non-linear overlap observed in the EDA, a mix of linear and non-linear models was selected for evaluation.

Logistic Regression (Baseline):

  - Used as a baseline to test if the classes are linearly separable. If this performs poorly, it confirms the need for complex boundaries.
  - Result file: submission_logreg_0.982.csv (Macro F1: 0.982)

Bayesian Logistic Regression:

  - A probabilistic extension of logistic regression, providing uncertainty estimates for predictions.
  - Result file: submission_bayesian_logreg_0.978.csv (Macro F1: 0.978)

Support Vector Machine (SVM):

  - Specifically with an RBF Kernel, chosen to handle non-linear decision boundaries by mapping the data into a higher-dimensional space.
  - Result file: submission_svm_0.978.csv (Macro F1: 0.978)

Random Forest Classifier:

  - An ensemble tree-based method chosen for its robustness to outliers and ability to model complex interactions without heavy preprocessing.
  - Result file: submission_randomforest_0.978.csv (Macro F1: 0.978)

XGBoost:

- Selected for its high performance on structured data and ability to learn complex decision surfaces through iterative correction of errors.
- Result file: submission_xgb_0.982.csv (Macro F1: 0.982)

LightGBM (LGBM):

- A gradient boosting framework that uses tree-based learning algorithms, known for its speed and efficiency.
- Result file: submission_lgbm_0.992.csv (Macro F1: 0.992)

Neural Network (NN):

- Used to capture highly non-linear relationships in the data, especially when feature interactions are complex.
- Result file: NNsubmission_0.988.csv (Macro F1: 0.988)

Discussion on the Performance of Different Approaches

- **Linear vs. Non-Linear:**
  - The relatively lower performance of Logistic Regression and Bayesian Logistic Regression compared to tree-based and neural models suggests that the classes are not linearly separable. Non-linear models like SVM, Random Forest, XGBoost, LGBM, and NN performed better, likely due to their ability to capture complex boundaries, especially for the central "Group_A" cluster.
- **Impact of Scaling:**
  - Scaling significantly improved the performance of KNN and SVM compared to unscaled versions, as these models are sensitive to feature magnitudes.
- **Best Model:**
  - LightGBM (LGBM) achieved the highest Macro F1 score of 0.992, effectively capturing the curved boundaries between Group_A and Group_B.Both the Neural Network (NN) and Support Vector Machine (SVM) models achieved strong results, with Macro F1 scores of 0.988 and 0.985 respectively. However, despite multiple attempts at hyperparameter tuning, further improvements in their scores could not be achieved. This suggests that these models may

    have reached their performance ceiling on this dataset, possibly due to feature limitations, cor very limited size of dataset.

- **Class Balance Handling:**
    - Some models struggled with the minority class (Group_A). The Macro F1 score reflects the ability to balance performance across all classes. Models with higher Macro F1 scores (LGBM, NN) managed to classify the minority class better, while linear models' scores were dragged down by misclassifying the smaller group.