CytoAutoCluster: Enhancing Cytometry with Deep Learning

This Jupyter Notebook focuses on preprocessing and analyzing a dataset titled `Levine_32dim.csv` to prepare it for machine learning tasks, including self-supervised learning for clustering. Below is a structured summary of the steps covered in the notebook:

1. Library Imports

- Essential libraries such as `pandas`, `numpy`, `seaborn`, and `matplotlib` for data manipulation and visualization are imported.

- The notebook also uses `keras` for building self-supervised models and `scikit-learn` for data preprocessing.

2. Dataset Loading and Initial Exploration

- The dataset `Levine_32dim.csv` is loaded from Google Drive.

- Columns and initial rows of the dataset are examined to understand the data structure.

3. Data Cleaning and Preprocessing

- Duplicate Removal: Duplicate rows are identified and counted.

- Handling Missing Values: Checks for missing values and conversion of non-numeric data to `NaN` are performed.

- Outlier Detection: Z-scores are calculated for outlier detection, and box plots visualize feature distributions, specifically focusing on `Cell_length`.

- Feature Range Calculation: The range (minimum and maximum values) of each feature is calculated to assess data variability.

4. Exploratory Data Analysis (EDA)

- Class Distribution: Analyzes the distribution of class labels to check for data balance.

- Correlation Analysis: A correlation matrix is generated and visualized with a heatmap to understand feature interrelationships.

5. Feature Scaling

- Standard scaling is applied to normalize the data for further processing.

6. Self-Supervised Learning Model Setup

- A self-supervised model is implemented with a binary mask function to perform feature masking.

- Hyperparameters (e.g., batch size, learning rate, and number of epochs) for the model are defined.

- The self-supervised model architecture is built using `Keras`, with hidden layers to encode and decode the input data.

7. Semi-Supervised Learning with Logistic Regression and XGBoost

- Further sections involve implementing `LogisticRegression` and `XGBoostClassifier` to predict labels for semi-supervised tasks, particularly for unlabeled inputs.

- Model performance is evaluated based on log-loss and other relevant metrics.

Summary of Actions Taken

The notebook performs extensive preprocessing, including duplicate handling, outlier analysis, correlation checks, and scaling. The self-supervised model aims to learn representations for clustering or classification, and semi-supervised learning is introduced to predict labels on unlabeled data. The document prepares the data comprehensively for machine learning, facilitating robust feature extraction and model training.