CytoAutoCluster: Enhancing Cytometry with Deep Learning

This Jupyter Notebook focuses on preprocessing, analyzing, and implementing self-supervised learning on a cytometry dataset to prepare it for clustering and classification tasks.

1. **Library Imports**
    - Essential libraries such as pandas, numpy, seaborn, and matplotlib are imported for data manipulation and visualization.
    - Machine learning libraries keras and scikit-learn are used to create self-supervised learning models and preprocess data.
2. **Dataset Loading and Exploration**
    - The cytometry dataset is loaded, and initial rows and columns are examined to understand its structure and content.
3. **Data Cleaning and Preprocessing**
    - Duplicate Removal: Duplicate rows in the dataset are identified and removed to ensure data quality.
    - Outlier Detection: Outliers are identified using Z-scores, with box plots used to visualize feature distributions.
    - Feature Range Calculation: Each feature's range is calculated, allowing analysis of data spread and variability.
    - Standardization: The dataset is scaled to standardize feature ranges for improved performance in downstream machine learning tasks.
4. **Data Corruption and Masking**
    - Binary Masking: A binary mask is applied to randomly corrupt certain features, generating corrupted inputs for self-supervised learning.
    - Data Shuffling: Feature values are randomly shuffled to create a noisy version of the dataset, adding variation for model training.
5. **Dimensionality Reduction Techniques**
    - PCA and t-SNE: Principal Component Analysis (PCA) and t-SNE are used to reduce dimensionality and visualize data in a lower-dimensional space for pattern detection and clustering.
6. **Self-Supervised Learning Model**
    - A self-supervised model architecture is implemented using an autoencoder structure to predict masked features and generate robust data representations.

- The model is trained on corrupted data, with binary cross-entropy and mean squared error losses optimized through the keras library.

7. **Semi-Supervised Learning with Logistic Regression and XGBoost**
    - The model includes LogisticRegression and XGBoostClassifier to label unlabeled data and predict outcomes.
    - Model evaluation includes performance metrics like log-loss to assess prediction accuracy and overall model efficacy.
8. **Variance Analysis and Statistical Calculation**
    - Variance Calculation: A table of standard variance values for each feature is created to assess feature variability across the dataset.

This project integrates advanced data preprocessing steps, such as outlier analysis and feature standardization, and explores dimensionality reduction techniques. Self-supervised and semi-supervised learning models are employed to generate meaningful representations and predict labels on unlabeled data, preparing the dataset for robust clustering and classification tasks.