<u>CytoAutoCluster:Enhancing Cytometry withDeep Learning</u>

<u>INTRODUCTION</u>:
In data-driven settings, accurately evaluating object features and categorising them into distinct groups is essential across many applications. CytoAutoCluster addresses this need by clustering cells into unique groups based on their characteristics. The goal is computational, with a focus on designing an effective clustering system that leverages semi-supervised learning techniques for improved efficiency.

<u>PROBLEM OVERVIEW</u>:
Clustering cytometry data poses significant challenges, primarily due to the following reasons:

1. **<u>High Dimensionality</u>**: Cytometry produces extensive, high-dimensional datasets that are challenging to visualize and interpret using traditional clustering approaches, such as k-means or hierarchical clustering. These methods struggle with the complex structure and variability within cytometric data.
2. **<u>Scarcity of Labels</u>**: The majority of cytometry data is unlabeled, limiting the ability to categorise cells accurately using supervised learning techniques. The difficulty in obtaining high-quality, labelled data in biological studies further complicates the task, making it challenging to define group assignments.
3. **<u>Noise and Variability</u>**: Cytometric data is often impacted by biological variability and noise, making it harder to process accurately. The high levels of noise, along with differences in biological features, reduce the effectiveness of standard clustering methods and demand more advanced techniques to identify relevant patterns.

<u>OBJECTIVES</u>:

The main goals of CytoAutoCluster are:

1. **<u>Create a Semi-Supervised Learning Framework</u>**: Develop an algorithm that effectively combines labeled and unlabeled data to boost clustering effectiveness in cytometric data analysis.

2. **Increase Clustering Accuracy**: Apply deep learning techniques to achieve higher accuracy in identifying complex cell groups compared to conventional methods.
3. **Minimize Labeling Needs**: Reduce the dependency on large labeled datasets by making greater use of unlabeled data, which cuts down on time and resources for data preparation.
4. **Enhance Result Interpretability**: Provide visualization tools that aid researchers in interpreting clustering outcomes and understanding the biological significance of identified cell groups.
5. **Ensure Scalability and Efficiency**: Develop an algorithm that is both scalable and efficient, allowing it to manage the large datasets typically generated in cytometry studies.

CODE IMPLEMENTATION :

- **Dataset Loading**:

  The process begins with loading the Levine CytOF 32-dimensional dataset into the environment. This data, saved in CSV format, is imported into Google Colab for further analysis.

- **Data Exploration**:

  After loading, the dataset is initially explored to understand its layout, such as by viewing the first few rows and determining the data's overall shape. This helps in identifying the number of features and records available for clustering analysis.

- **Null Value Analysis**:

  Since semi-supervised learning is heavily reliant on data completeness, identifying null values is essential. This involves a detailed check of the dataset to locate columns with missing values and specific rows affected, which is vital for evaluating data quality.

- **Comparing Null and Non-Null Values**:

  To better understand the dataset's structure, visualisations are generated to display the spread of null and non-null values. These graphical views provide insights into which features have missing data and the extent of this impact, guiding data preprocessing steps.

- **<u>Data Analysis Methods</u>**:
    1. Histograms: Show the distribution of values within each feature.
    2. Correlation Matrix: Used to evaluate relationships among features, assisting in feature selection.
- **<u>Correlation Matrix</u>**:

  A correlation matrix is constructed to reveal relationships between features:

    1. Positive Correlations: Both features increase or decrease in tandem.
    2. Negative Correlations: One feature increases while the other decreases.
    3. Unrelated Features: Features with minimal correlation can be useful for dimensionality reduction. Analyzing these relationships aids in selecting important features and reduces redundancy by eliminating highly correlated ones, simplifying the model.
- **<u>Feature Value Range</u>:**

The range of values for each feature is calculated, providing insights into the scale and variability of each attribute.

- **<u>Box Plot Analysis</u>**:

Box plots are created to visually assess the distribution of each feature:

1. Median and Quartiles: Show data distribution centered around the median, indicating spread and skewness.
2. Outliers: Outliers beyond the interquartile range are highlighted, informing feature engineering or data cleaning decisions.

- **<u>Skewness and Kurtosis Analysis</u>**:

1. **<u>Skewness</u>**: Measures the asymmetry of the feature distributions:

   a)Positive Skew: Distribution has a longer tail on the right.

   b)Negative Skew: Distribution has a longer tail on the left.

2.**<u>Kurtosis</u>**: Measures the "tailedness" or the peak of the distribution:

   a)High Kurtosis: Leptokurtic, with sharp peaks and heavy tails.

   b)Low Kurtosis: Platykurtic, with flatter distributions.

This analysis identifies features that may require transformation to improve normality and stabilise variance, aiding machine learning algorithms.

- **Dimensionality Reduction Techniques**:

  1. **Principal Component Analysis (PCA)**:

  PCA reduces dimensionality by retaining essential features that account for the majority of data variance. It is particularly useful in simplifying high-dimensional datasets for easier visualisation and analysis.

a)Variance Explained: Calculates the significance of each component to capture at least 95% of variance in fewer dimensions.

b)Transformation to Lower Dimensions: This reduces computational complexity and overfitting risks, improving model efficiency.

2. **t-Distributed Stochastic Neighbour Embedding (t-SNE)**:

t-SNE is mainly used for visualising data by mapping it into 2D or 3D space, uncovering underlying patterns and clusters.

a)Preserving Local Similarities: Unlike PCA, t-SNE captures non-linear relationships by focusing on local structures instead of global ones.

b)Cluster Visualisation: Reveals potential groupings within the data, especially useful in semi-supervised learning for understanding unlabeled data structure.

- **Autoencoders**:

  Autoencoders are neural networks that perform feature extraction in semi-supervised learning, making use of both labelled and unlabeled data.

- **Autoencoder Process**:

1. Encoding: Compresses data into latent features.
2. Latent Space: Stores reduced-dimensional data.
3. Decoding: Reconstructs original data from latent space, minimising dimensionality while preserving essential information.

- **Applications and Data Requirements:**

1. Input Dimensions: Autoencoders process high-dimensional cytometry data to produce lower-dimensional representations.

2. Noise Introduction: Artificial noise or dropout can be applied to input data to enhance feature extraction.

- **Binary Masking and Data Augmentation** :

To enhance the robustness of the clustering model, binary masking and data augmentation techniques are applied:

1. **Binary Masking**: This technique is used to simulate missing data by randomly masking 30% of feature values in the dataset. By training on partially obscured data, the model learns to handle incomplete information, improving its resilience in real-world applications.
2. **Shuffling Feature Columns**: To further introduce variability and prepare the data for semi-supervised clustering, feature columns are shuffled. This process helps to create diverse training samples, allowing the model to generalize better and handle a wider range of data patterns.

- **Data Splitting and Label Handling:**
  After corrupting data, a mask identifies "labeled" (corrupted) and "unlabeled" (original) entries, allowing for the creation of a semi-supervised framework. Data is split into training and testing sets, with labeled data used for training and uncorrupted data as the target for reconstruction. If no dedicated label column exists, a new label column or DataFrame index serves as a label.

- **Potential Downstream Tasks:**
  The latent representations learned by the autoencoder can support various downstream tasks:

1. **Denoising**: Removing noise to restore original data features.
2. **Anomaly Detection**: Identifying unusual data points based on reconstruction errors.
3. **Dimensionality Reduction**: Using the latent space as a low-dimensional embedding for classification or clustering.

- **<u>Logistic Regression and XGBoost with Log Loss</u>:**
  To further analyze clustering performance, logistic regression or XGBoost can be applied to the autoencoder's reduced-dimensional data, with log loss as an evaluation metric. Log loss penalizes incorrect predictions, providing a basis for comparing models, tuning parameters, and improving clustering accuracy.