CAS ADS Final Project Report: Multilabel classification models of cell type and condition

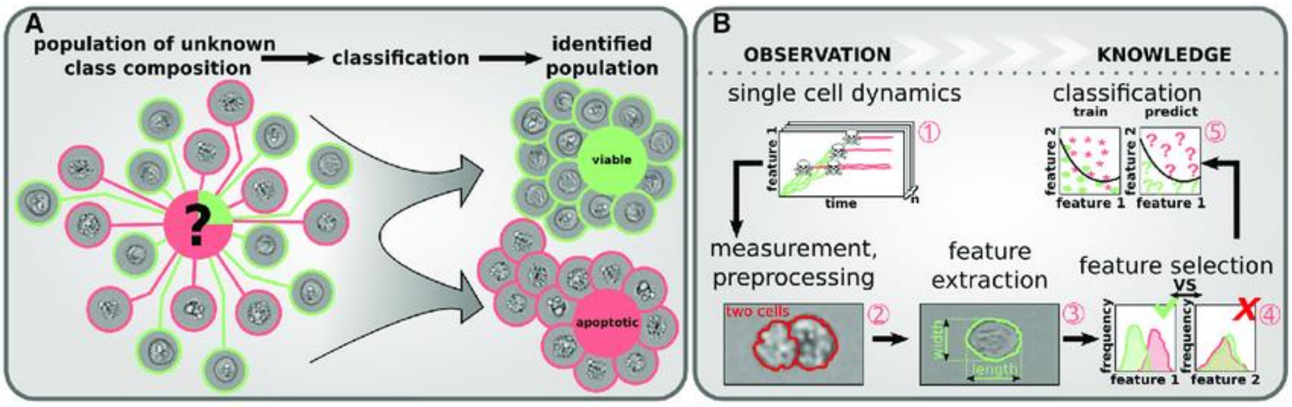

Figure 1:Exemple of use of Data science for flow cytometry data[1]

Author: Romain Topalian (topalian@hochschule-bc.de)

Abstract

This study presents a novel approach for cell state classification in imaging flow cytometry data. Machine learning algorithms and feature extraction techniques have been used to extract meaningful information from multidimensional imaging data. Our approach utilizes spatial and morphological features captured by imaging flow cytometry for accurate discrimination of different cell states and types. By employing UMAP for dimension reduction, data were pre-processed. A clustering DBSCAN algorithm allows the labelling of each cell. Various machine learning algorithms, including support random forests, and MLP models, are used for cell classification based on the previous labelled data. Evaluation using flow cytometry datasets demonstrates the effectiveness in accurately classifying cell states and types.

Inhalt

Figures

Tables

Abbreviations

UMAP	Uniform Manifold Approximation and Projection
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
SVM	Support Vector Machine
MLP	Multi layer perceptron
PI	Propidium iodide
LDA	Linear Discriminant Analysis
kNN	k-Nearest Neighbour
MLP	MultiLayerPerceptron
ROC	Receiver Operating Characteristic
AUC	Area Under the ROC curve

I. Introduction

Cell state classification plays a crucial role in understanding cellular heterogeneity. Flow cytometry enables high-throughput analysis of cellular features at the single-cell level, providing valuable insights into cell states and types. In this study, a novel approach for cell state classification in flow cytometry data is proposed. Machine learning algorithms and feature extraction techniques are utilized to extract meaningful information from the multi-dimensional data. The approach leverages the rich spatial and morphological features captured by imaging flow cytometry to accurately discriminate between different cell states and types. The flow cytometry data is first preprocessed through dimension reduction, normalization, and feature extraction. Subsequently, cells are grouped and labeled using a clustering algorithm. Various machine learning algorithms, such as random forests, decision trees, and deep learning models, are then employed to classify cells into different states and types based on the extracted features. The performance evaluation demonstrates the effectiveness and robustness of the approach in accurately classifying cell states and types. The results highlight the potential of the proposed methodology in biomedical research, drug discovery, and clinical applications, providing valuable insights into cellular heterogeneity at the single-cell level.

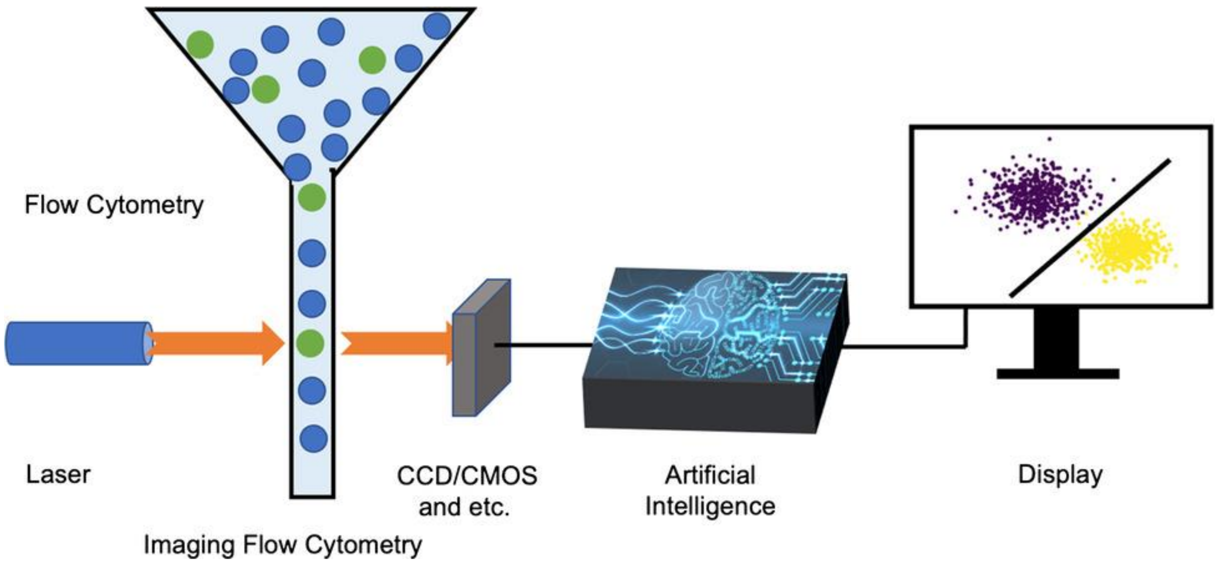

Figure 2: Overview of intelligent imaging flow cytometry.[2]

II. Data

1) Raw data

The raw data used in this study were obtained from a flow cytometer in mqd format. To facilitate analysis, the data were converted to fcs format using the MacsQuantify software. Subsequently, a Python library was employed to process and analyze the data. The resulting dataframe consists of 15 variables: HDR-T, HDR-CE, HDR-SE, HDR-V, FSC-A, FSC-H, FSC-W, SSC-A, SSC-H, SSC-W, VioBlue-A, VioBlue-H, VioBlue-W, Calcein-A, Calcein-H, Calcein-W, PI-A, PI-H, and PI-W. Each variable corresponds to a specific measurement or parameter associated with the individual cells. These variables capture information such as size (FSC), granularity (SSC), fluorescence intensity in the VioBlue channel, as well as the fluorescence intensity of Calcein and PI dyes. The resulting dataframe represents a 15-dimensional matrix that serves as the basis for further analysis in this study.

2) Data flow

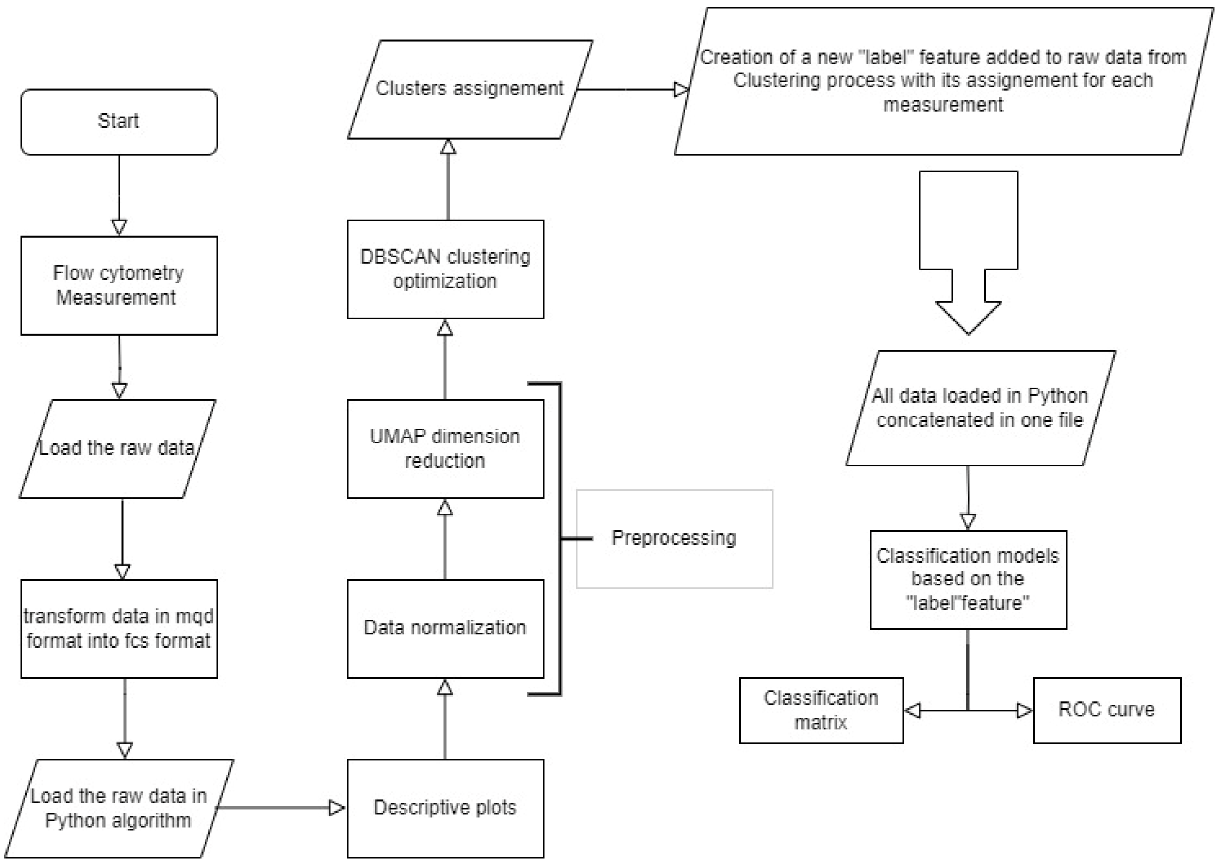
Start
Flow cytometry Measurement
Load the raw data
transform data in mqd format into fcs format
Load the raw data in Python algorithm
Descriptive plots
Data normalization
UMAP dimension reduction
DBSCAN clustering optimization
Clusters assignement
Preprocessing
Creation of a new "label" feature added to raw data from Clustering process with its assignement for each measurement
All data loaded in Python concatenated in one file
Classification models based on the "label"feature"
Classification matrix
ROC curve

Figure 3: Data flow of this project

3) Descriptive data analysis

a) Correlation matrix

The data can be visualized with this correlation matrix:

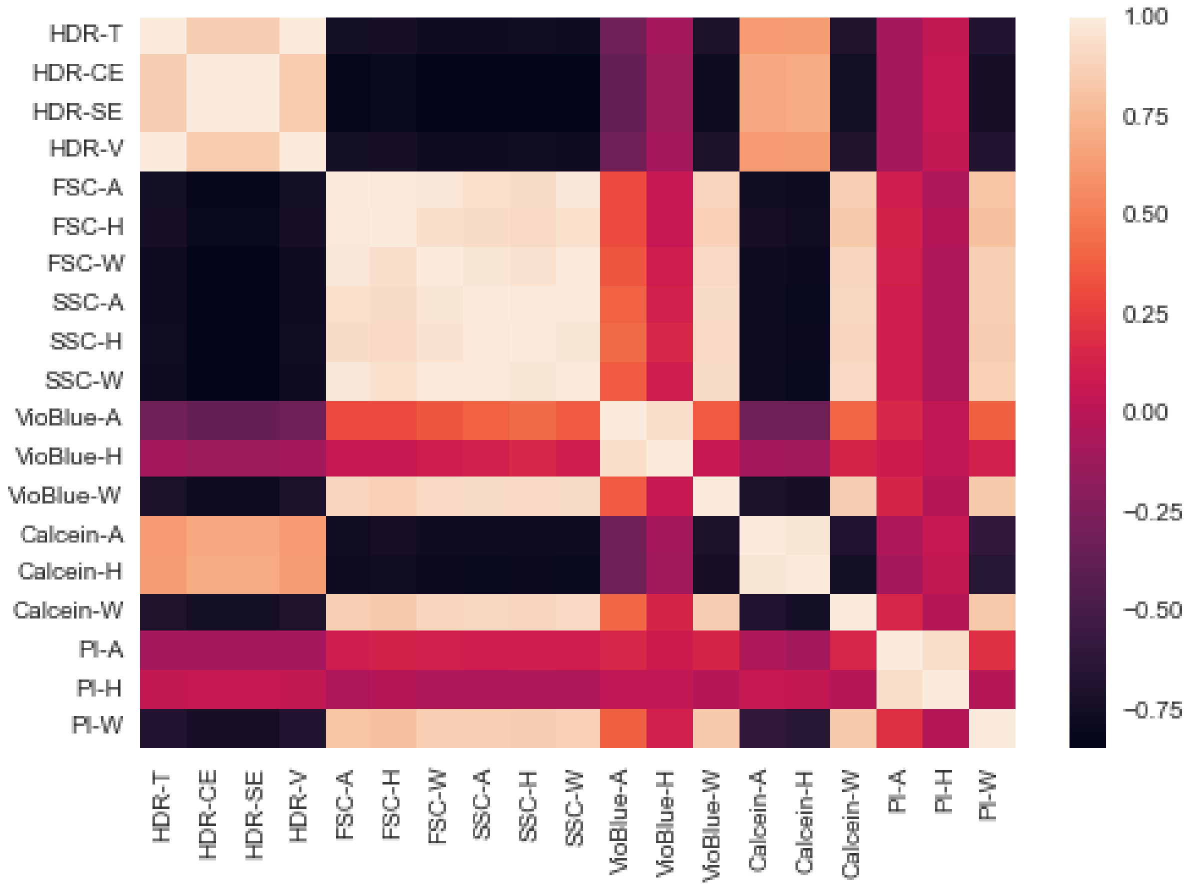

Figure 4:Correlation matrix of one measurement

The variables HDR, which represent the time when the measurements were taken, are not deemed significant and will be removed during the preprocessing phase. These variables are not expected to contribute to the classification task or provide meaningful insights into the cell type and state. By removing them during the preprocessing step, the dataset can be simplified and focused on the relevant features that have a more direct impact on the classification process. This allows to streamline the analysis and improve the efficiency and effectiveness of the classification models.

b) Distribution

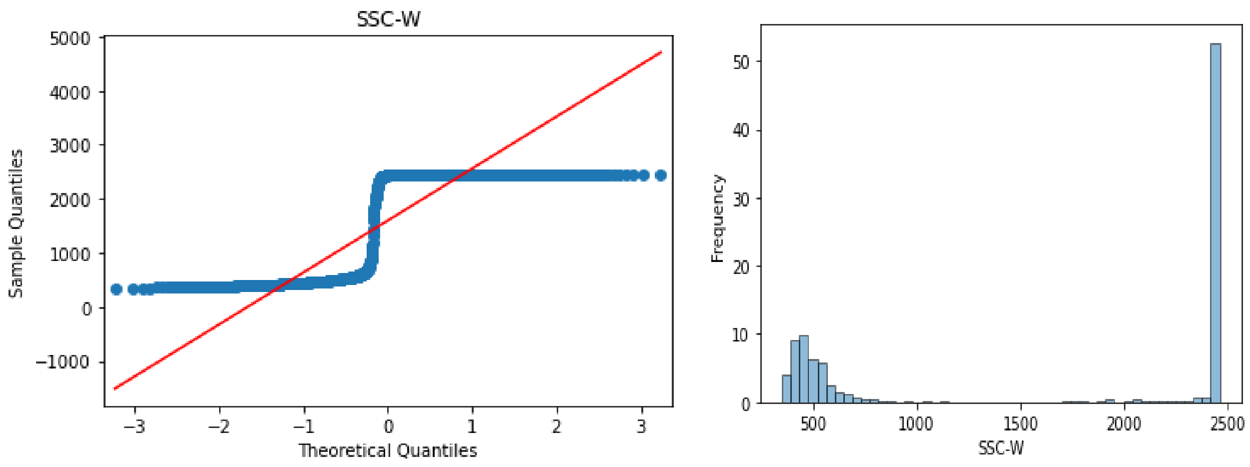

Figure 5: QQ-plot (left) and frequency (right) of the data distribution of the variable SSC-W.

In this case, the QQ-plot shows a noticeable deviation, suggesting that the SSC-W variable does not follow a normal distribution. It is the case for all variables.

c) Descriptive plots

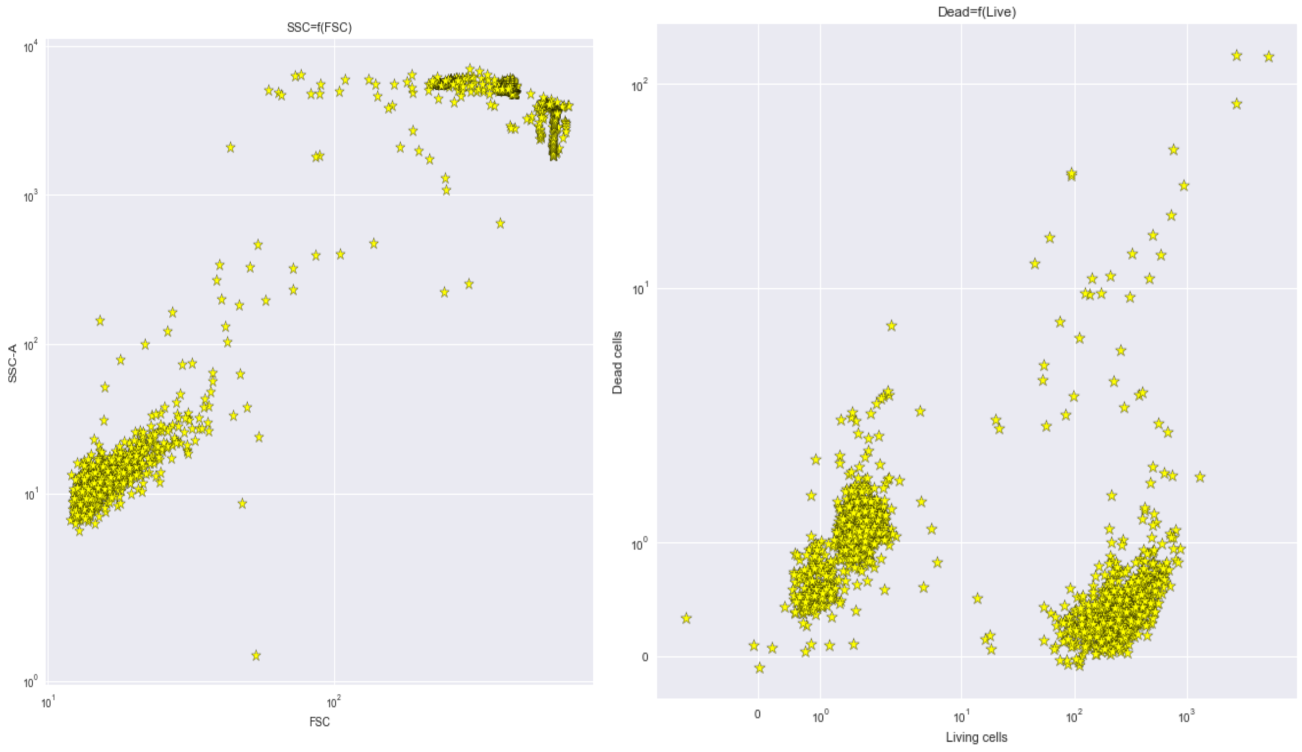

Figure 6:On the left, scatter plot of FSC-A (cell size) against SSC-A (cell granularity). On the right, scatter plot of Calcein-A (Living cells) against PI-A (dead cells). Each point represents one measurement, which means one cell.

These descriptive plots offer a visual representation of the cellular composition within the measurement. They allow to observe the diversity and distribution of cell types based on their size and granularity, as well as assess the relative proportions of living and dead cells. These insights can guide further analysis and classification processes, enabling a deeper understanding of cellular heterogeneity and viability.

Overall, these descriptive plots provide a preliminary overview of the cell populations and viability status present in the imaging flow cytometry data, serving as valuable exploratory tools for initial data exploration and interpretation.

III. Clustering analysis

1) Preprocessing

The preprocessing of the data in this study involved two main steps.

The first step involved normalizing the selected variables: FSC-A, FSC-H, FSC-W, SSC-A, SSC-H, SSC-W, VioBlue-A, VioBlue-H, VioBlue-W, Calcein-A, Calcein-H, Calcein-W, PI-A, PI-H, and PI-W. The StandardScaler() library was employed for this purpose. Normalization ensures that the variables are on a comparable scale, removing any potential bias caused by differences in their original units or magnitudes. This step allows for fair comparison and analysis of the variables in subsequent analyses.

The second step of the preprocessing involved dimension reduction using UMAP (Uniform Manifold Approximation and Projection). UMAP is a dimensionality reduction technique that aims to preserve the underlying structure and relationships of the data while reducing its dimensionality. By projecting the data onto a lower-dimensional space, UMAP enables a better separation of the data points, improving the effectiveness of subsequent clustering processes. This step helps to capture the essential information of the data while reducing computational complexity and potential noise or redundancy in the high-dimensional space.

Overall, the combination of normalization and dimension reduction using StandardScaler() and UMAP, respectively, ensures that the data is appropriately prepared for subsequent analysis, such as clustering. It facilitates unbiased comparison, enhances data separation, and reduces the computational burden, ultimately leading to more accurate and meaningful results in the subsequent steps of the study.

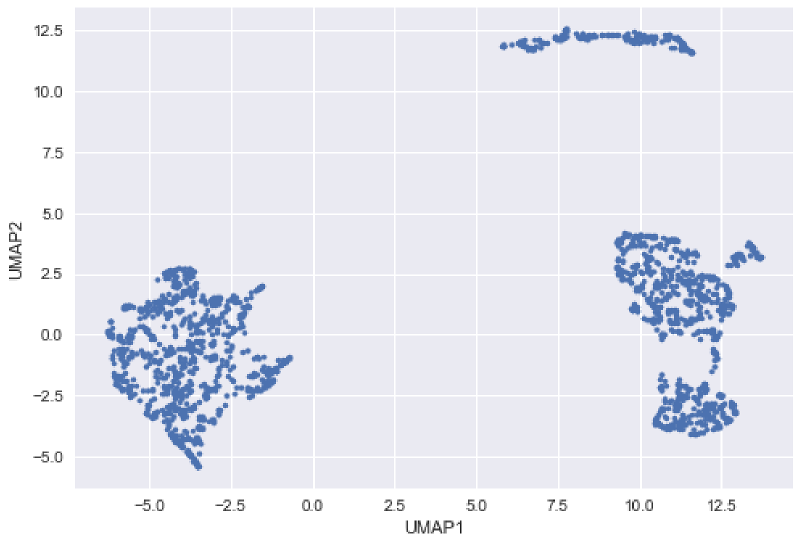

Figure 7: UMAP of raw data: reduction of a 15-dimensional matrix to a 2-dimensional matrix

2) Clustering DBSCAN

In this study, the data undergo clustering analysis using the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm. The DBSCAN algorithm is applied to identify groups in the data based on their proximity and density in the feature space. By considering the density of neighboring points (min point is the value representing the minimum points to form a cluster) and using a distance threshold (epsilon value), the algorithm determined whether a point belonged to a cluster or was considered noise.

a) Hyperparameters tuning

To tune the hyperparameters for the DBSCAN algorithm (min points and epsilon value), a loop is performed over a wide range of parameter values. The goal is to identify the optimal combination that maximizes the silhouette score, which measures the quality of clustering.

The loop iterates through various combinations of min points and epsilon values within the defined range. For each combination, the DBSCAN algorithm is applied to the preprocessed data. The resulting clusters are evaluated using the silhouette score, which quantifies how well each data point fits within its assigned cluster compared to other clusters.

By calculating the silhouette score for each combination of hyperparameters, the loop identifies the combination that yields the highest score. This combination represents the optimal configuration for the DBSCAN algorithm, resulting in the best clustering performance for the given dataset.

The silhouette score serves as a valuable criterion for selecting the best combination of hyperparameters, as it considers both the cohesion within clusters and the separation between different clusters. A higher silhouette score indicates more distinct and well-separated clusters, reflecting a better representation of the underlying structure in the data.

Overall, the process of tuning the hyperparameters using the silhouette score allows for the identification of the optimal combination that maximizes the quality of clustering in the DBSCAN algorithm. This ensures that the subsequent analysis and interpretation of the clustering results are based on the most accurate and reliable configuration (Figure 8).

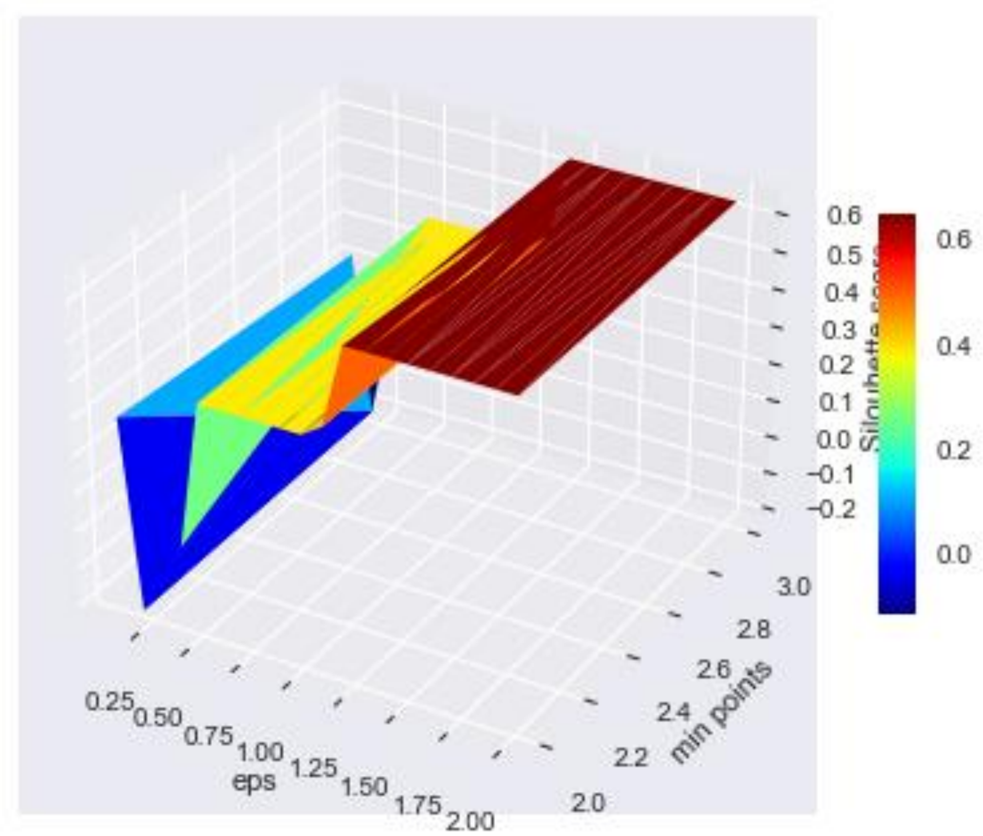

Figure 8: 3D-plot of the silhouette score against min points and epsilon value

b) Cluster identification

After tuning the hyperparameters, a DBSCAN was performed using the optimal parameter values. This allowed for the identification of the number of clusters present in the data, as well as the detection of noise points. Additionally, the characteristics of each cluster were determined through the clustering analysis.

By utilizing the optimal parameter values, the DBSCAN algorithm effectively grouped the data points into clusters based on their proximity and density. The resulting clusters provided valuable insights into the underlying structure and patterns within the dataset. The number of clusters was determined automatically by the algorithm, without requiring prior knowledge or assumptions.

Furthermore, the identification of noise points helped identify data points that did not belong to any specific cluster and were considered outliers. This information is crucial in distinguishing significant data points from potential artefacts or background noise.

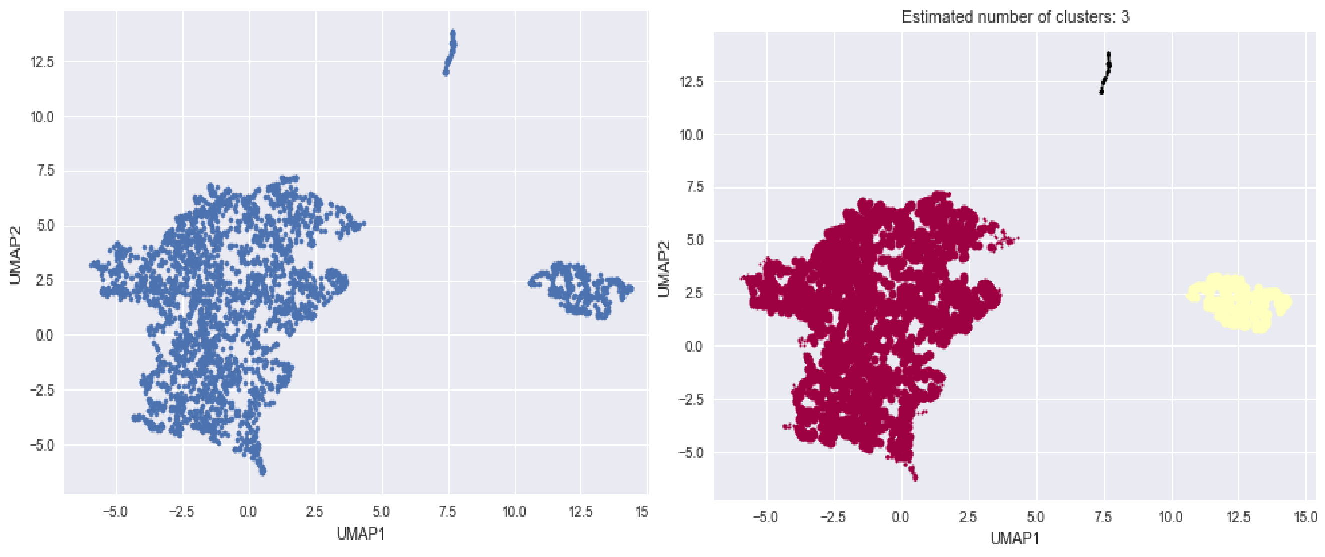

Figure 9:On the left, UMAP plot before the DBSCAN clustering. On the right, UMAP plot after the identification of clusters (three different clusters in this case) after DBSCAN clustering. The black points are the noise points.

c) Cluster assignment

In the determination of cluster assignments, the experimentator plays a crucial role based on the characteristics of each cluster. Thresholds for specific features are established through control experiments. These thresholds are used to assign each cell in the measurement to a specific cell type (such as RE cells, OE cells, Trachea cells, HEK, CHO, E.Coli, CGB, Monocyte) or their conditions (Living cells, Dead cells, Apoptotic cells).

For instance, the threshold for the PI value, which indicates dead cells, is determined through measurements of cells treated with 30% Isopropanol. Generally, a higher PI value suggests a higher likelihood of the cluster corresponding to dead cells. Similarly, for living cells, the Calcein value is utilized to identify the living cell cluster. These thresholds are also specific for each type of cell.

To establish the ground truth for cell types, each type is individually measured to create a reliable reference. During the assignment process, a new feature called "label" is added to the raw dataframe, indicating the type of cell and its specific condition for each measurement. The calculated UMAP 1 and 2 are also added in the raw dataframe but won´t be considered during the classification process.

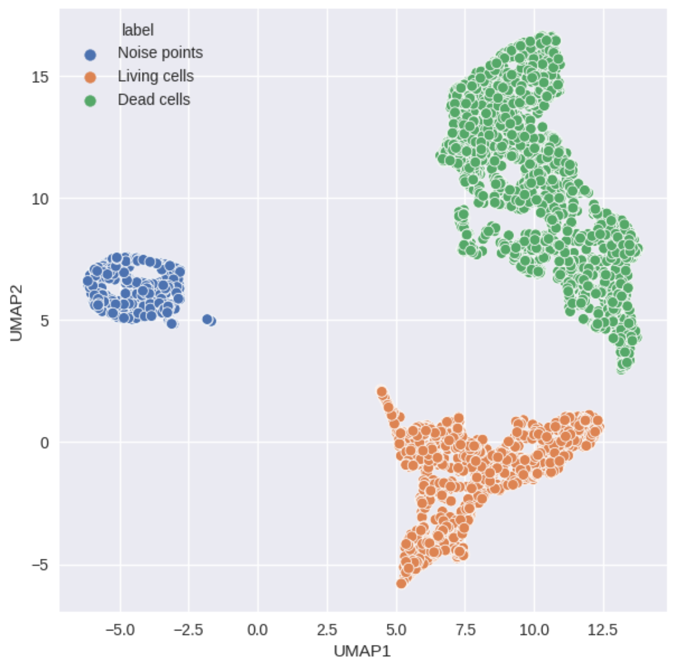

Figure 10: Cluster assignment for RE cells type. UMAP plot shows in green the dead cells, in orange living cells and in blue noise points.

Table 1: Raw dataframe with the "label" feature added after DBSCAN clustering assignment. This dataframe will be used for classification process.

	FSC-A	FSC-H	FSC-W	SSC-A	SSC-H	SSC-W	VioBlue-A	VioBlue-H	VioBlue-W	Calcein-A	Calcein-H	Calcein-W	PI-A	PI-H	PI-W	UMAP1	UMAP2	label
4068	19.009203	16.666172	570.293030	19.861055	14.041887	707.207458	0.703903	0.515667	682.517029	17.545357	14.886036	589.322693	77.031708	87.320381	441.086639	8.615017	10.556864	RE cells: Dead
4069	20.672577	25.955667	398.228546	4.850778	6.171317	393.009949	-0.044167	0.252510	-87.455086	8.792954	12.661298	347.237488	58.278889	69.504105	419.247803	10.450683	16.432312	RE cells: Dead
4070	87.264519	60.834488	717.229004	36.831543	24.297266	757.935974	1.223329	0.753969	811.259705	561.308533	550.971497	509.380768	72.258316	75.256554	480.079865	13.451611	8.076166	RE cells: Dead
4071	53.532436	45.670513	586.072205	25.613970	21.237614	603.033142	0.444918	0.501342	443.727783	13.282900	10.821662	613.718079	64.229744	67.460991	476.050964	10.525159	11.910025	RE cells: Dead
4072	20.092020	27.786844	361.538361	1.751410	2.193726	399.186279	-0.207676	0.234125	-443.515839	-1.438773	0.078224	-9196.487305	-0.199708	0.457114	-218.444931	5.460568	-5.160619	RE cells: Live

In conclusion, the determination of cell conditions is carried out by the user, as the thresholds are defined by the user based on control experiments. The clustering step helps create an accurate ground truth for the subsequent classification phase, facilitating the identification and characterization of different cell types and conditions.

IV. Classification models

After the clustering process, each measurement has been labeled with the cell type and state. The next step is to develop a classification tool capable of predicting the cell type and state directly from new measurements. This aims to provide insights into the different types of cells present in a measurement and assess the cell viability for each cell type.

Several classification models were tested in this study, including Naïve Bayes, Decision Tree, Random Forest, MultiLayer Perceptron, LDA, KNN, Logistic Regression, and AdaBoost. Each model underwent the following steps:

- Concatenation of all labeled raw dataframes and removal of the "UMAP1" and "UMAP2" columns.
- Division of the dataframe into two parts: the training set (80% of the data) and the test set (20% of the data). The "label" feature serves as the target variable for cell type and state prediction.
- Hyperparameter tuning of each model using the GridSearch() library.
- Building the model and displaying the confusion matrix (Figure 11), which provides an overview of the model's performance, and the classification report (Table 2), which includes precision, recall, F1-score, and support for each class.

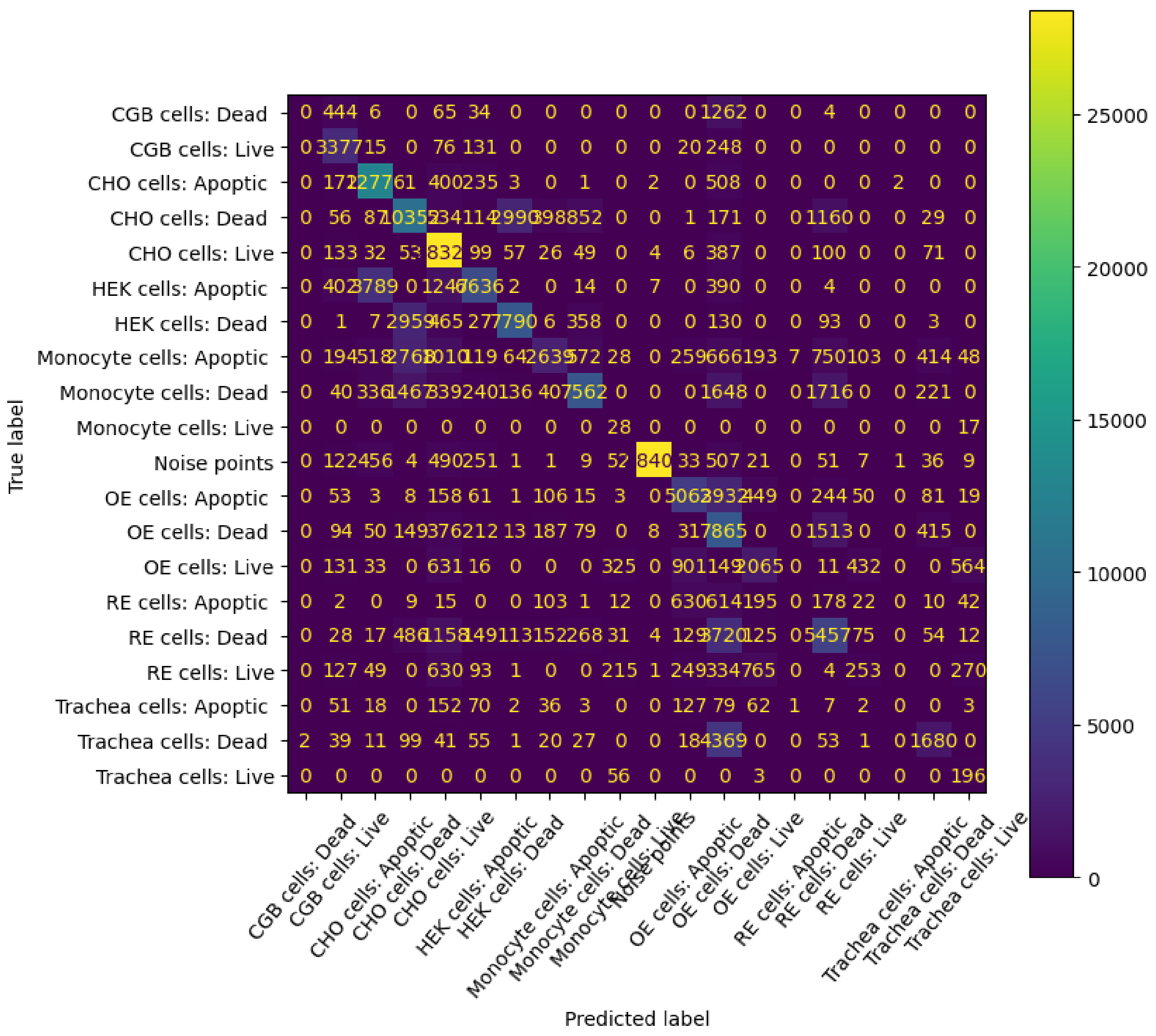

Figure 11: Confusion matrix of the LDA classification model

Table 2: Classification report of the LDA classification model

	precision	recall	f1-score	support
CGB cells: Dead	0.00	0.00	0.00	1815
CGB cells: Live	0.62	0.87	0.72	3867
CHO cells: Apoptic	0.70	0.91	0.79	14100
CHO cells: Dead	0.56	0.62	0.59	16744
CHO cells: Live	0.78	0.97	0.87	29339
HEK cells: Apoptic	0.78	0.53	0.63	12491
HEK cells: Dead	0.70	0.66	0.68	11839
Monocyte cells: Apoptic	0.71	0.25	0.38	10352
Monocyte cells: Dead	0.77	0.55	0.64	13745
Monocyte cells: Live	0.04	0.62	0.07	45
Noise points	1.00	0.93	0.96	30456
OE cells: Apoptic	0.68	0.55	0.61	9246
OE cells: Dead	0.30	0.72	0.43	10992
OE cells: Live	0.53	0.39	0.45	5258
RE cells: Apoptic	0.00	0.00	0.00	1833
RE cells: Dead	0.48	0.46	0.47	11978
RE cells: Live	0.27	0.08	0.13	2991

Trachea cells: Apoptic	0.00	0.00	0.00	613
Trachea cells: Dead	0.56	0.26	0.36	6416
Trachea cells: Live	0.17	0.77	0.27	255
accuracy			0.67	194375
macro avg	0.48	0.51	0.45	194375
weighted avg	0.69	0.67	0.66	194375

Additionally, the developed classification models underwent further evaluation and validation steps. K-fold stratified cross-validation was employed to assess the model's performance and generalization ability. This technique divides the dataset into K equally sized folds while ensuring that each fold maintains a similar distribution of cell types and states as the original dataset.

Furthermore, learning curves were plotted (Figure 12) to visualize the model's learning progress and evaluate its training and testing accuracy as the size of the training set varied. These curves provide insights into the model's capacity to learn from the data and its potential for overfitting or underfitting.

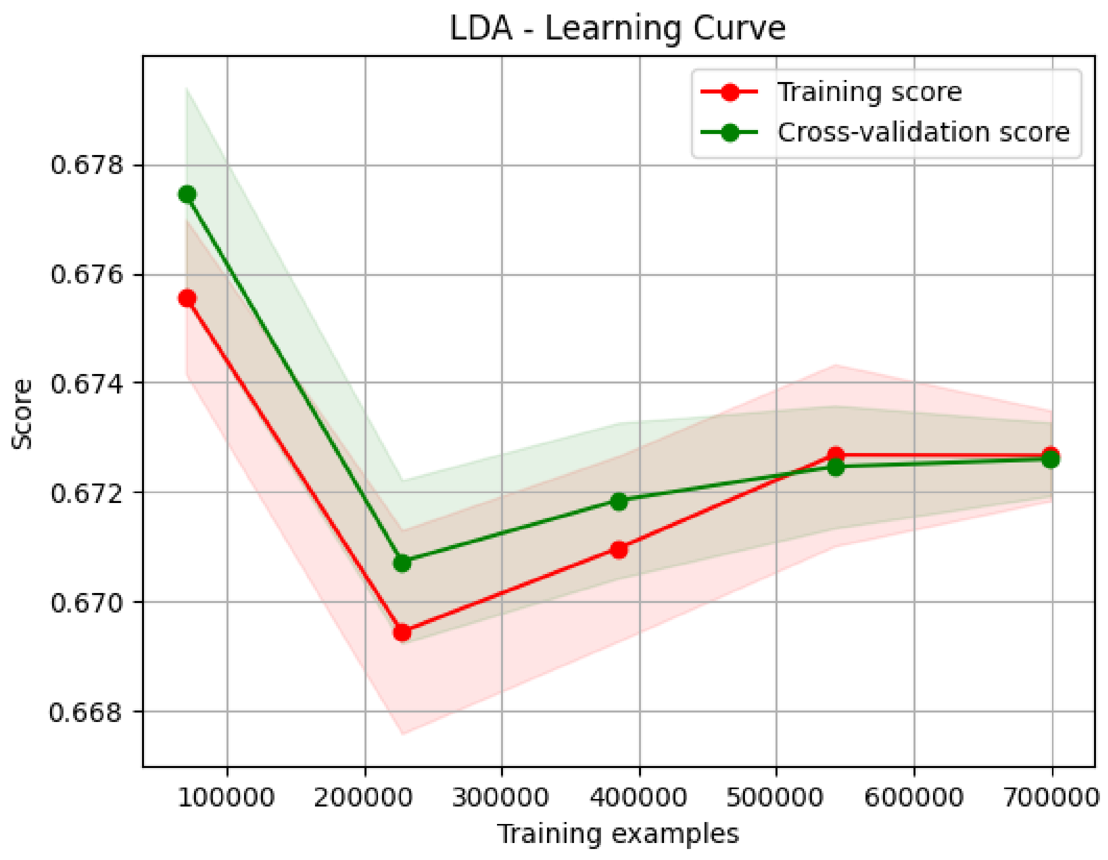

Figure 12:Learning curve of the LDA model classification

Another approach to evaluating the classification model is by plotting the micro-average One-vs-Rest Receiver Operating Characteristic (ROC) curve. The micro-average ROC curve provides a comprehensive assessment of the model's performance across all classes by aggregating the true positive rate (sensitivity) and the false positive rate (1-specificity) across all classes.

By plotting the micro-average ROC curve, the model's ability can visually be assessed to discriminate between different cell types and states. The curve demonstrates the trade-off between the true positive rate and the false positive rate, allowing us to determine the model's overall classification performance.

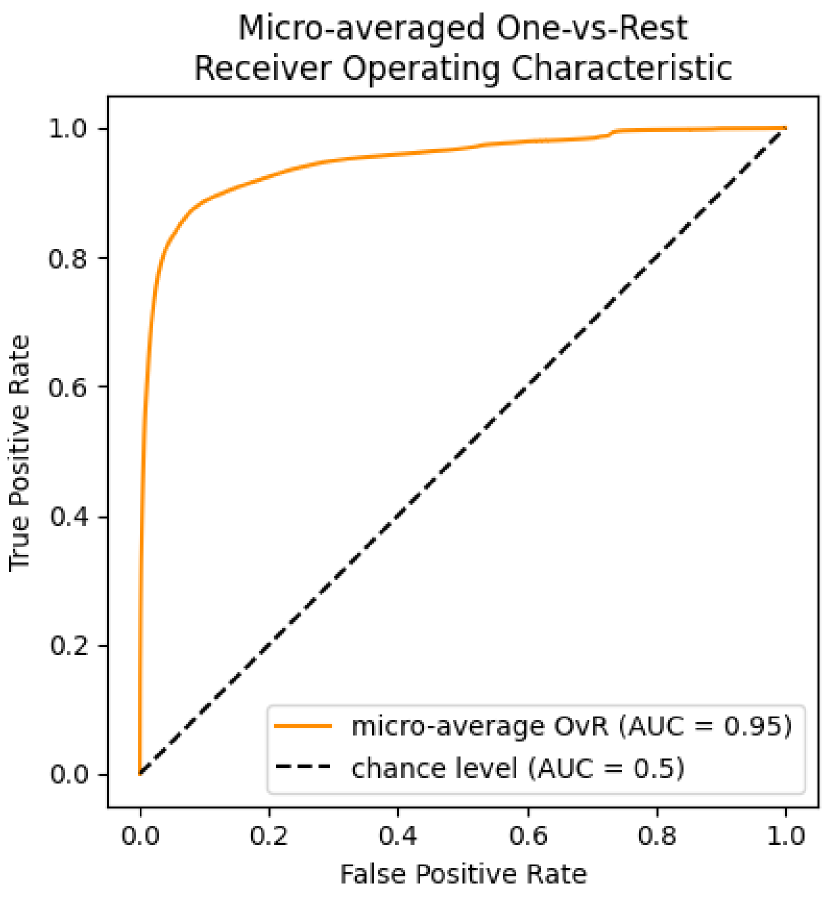

Figure 13:Micro-averaged One-vs-Rest ROC curve for LDA classification model

By following these steps for each classification model, the study aimed to evaluate their performance and determine the most suitable model for accurately predicting cell type and state based on the available features.

V. Results/Discussion

In order to assess and compare the performance of the classification models, their accuracies have been compared. Accuracy is a commonly used evaluation metric that measures the proportion of correctly classified samples out of the total number of samples.

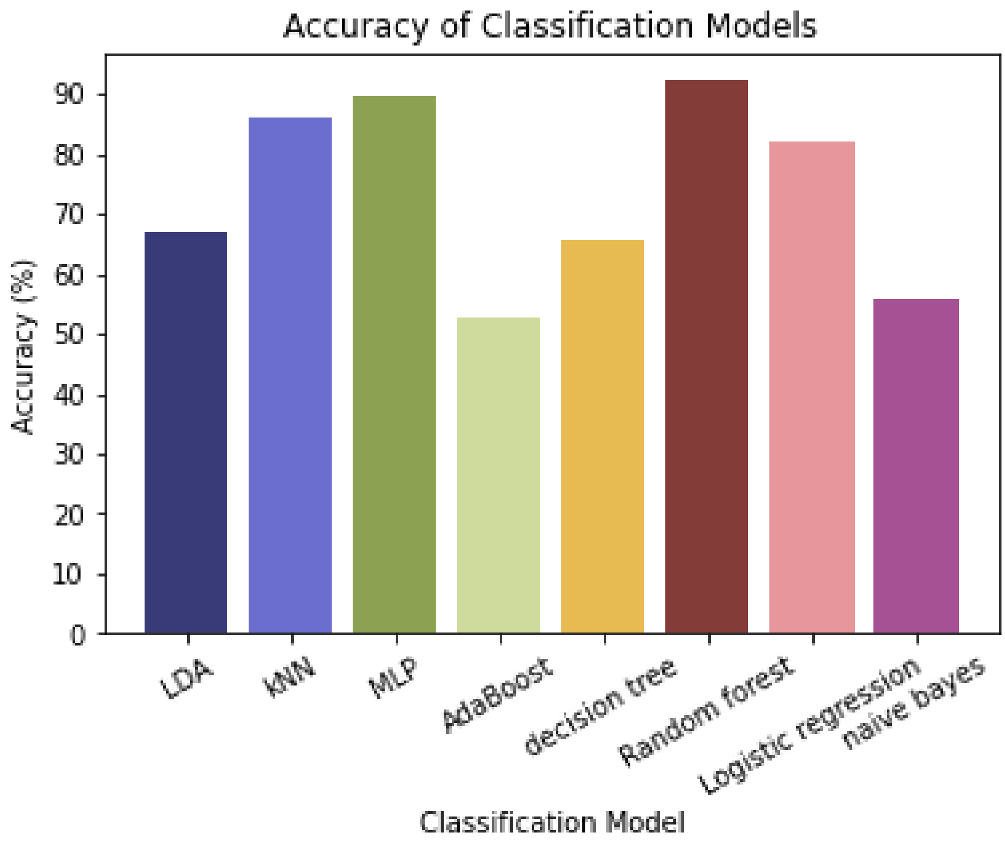

Figure 14: Accuracy of test dataset for different classification models

The Random Forest classification model achieved the highest accuracy of 92.13% among the tested models. However, it is crucial to evaluate whether this high accuracy is indicative of overfitting, where the model may be overly specialized to the training data and fail to generalize well to new, unseen data.

In the case of the Random Forest classifier, the training dataset accuracy of 100% suggests a potential overfitting issue. This high accuracy indicates that the model has memorized the training data, capturing even the noise or random variations present in the dataset. Consequently, there is a risk that the model may not perform as well on unseen data, limiting its usefulness for real-world applications.

On the other hand, the MLP (MultiLayer Perceptron) classifier exhibited an accuracy of 90.77% for the training dataset and 90.63% for the test dataset. These results indicate that the model is performing consistently well on both the training and test data, with a minimal difference in accuracy. This suggests that the MLP classifier is less prone to overfitting and has a better ability to generalize to new data.

Furthermore, the learning curve of the MLP classifier (Figure 15) demonstrates a reliable cross-validation score, which indicates that the model's performance remains consistent across different subsets of the data. This strengthens the confidence in the model's effectiveness and generalization capability.

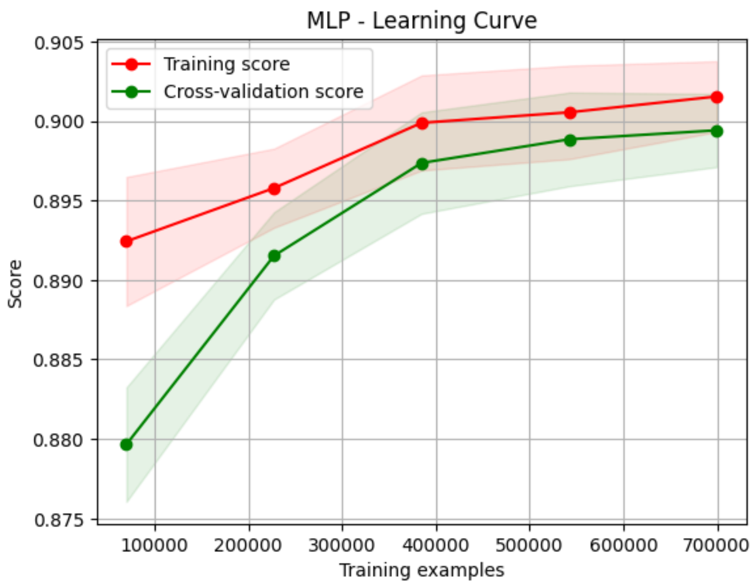

Figure 15: Learning curve of the MLP classifier

In the case of the MLP (MultiLayer Perceptron) model, the ROC curve for the One vs Rest multiclass classification problem (Figure 16) exhibits good results, with a minimum Area Under the Curve (AUC) of 0.9. The AUC is a summary measure of the ROC curve and represents the overall performance of the model. An AUC value of 0.9 indicates a high level of discrimination, with the model exhibiting a strong ability to correctly classify different classes.

The minimum AUC value of 0.9 suggests that the MLP model performs well across the different classes, maintaining a high level of sensitivity while keeping the false positive rate relatively low. This indicates that the model can effectively distinguish between the different classes, leading to reliable and accurate predictions.

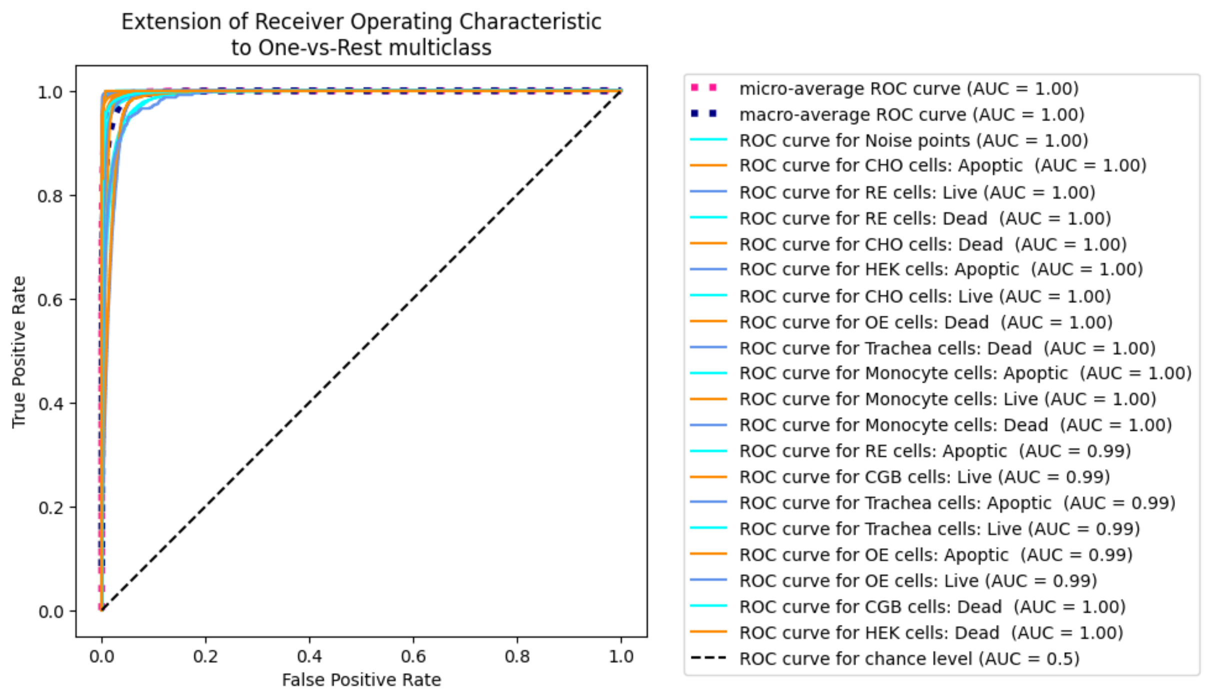

Figure 16: ROC curves for MLP classifier with AUC score

Conclusion

This work presented a comprehensive approach for cell state classification in imaging flow cytometry data. Starting with the preprocessing stage, the raw data obtained from the flow cytometer were transformed into a dataframe, and normalization techniques were applied to ensure consistent and comparable features across the dataset. Subsequently, dimension reduction using UMAP was employed to enhance the separation of data points and improve the clustering process.

The clustering stage utilized the DBSCAN algorithm to identify groups of cells and assign them labels based on their characteristics. This clustering step facilitated the creation of a reliable ground truth for cell type and state assignment, enabling subsequent classification.

The classification stage involved the evaluation of various machine learning algorithms, including Naïve Bayes, Decision Tree, Random Forest, MultiLayer Perceptron, LDA, KNN, Logistic Regression, and AdaBoost. These models were trained and tested using a labeled dataset, and their accuracies were compared to identify the most suitable algorithm for cell type and state prediction.

The overall findings of this work demonstrate the efficacy of the proposed methodology in accurately classifying cell states in imaging flow cytometry data. By leveraging advanced preprocessing techniques, clustering algorithms, and machine learning models, valuable insights into cellular heterogeneity can be obtained. The classifier model MLP achieved a high accuracy, indicating its potential for predicting cell types and states in new, unseen data.

In conclusion, the combination of preprocessing, clustering, and classification techniques presented in this work provides a comprehensive workflow for analyzing imaging flow cytometry data and offers promising avenues for advancing our understanding of cellular dynamics and mechanisms.

The code can be found at https://github.com/toutouni735/CAS-ADS-Final-project.

Acknowledgements

The author would like to express his sincere gratitude to CAS ADS who contributed to the successful completion of this study. I would like to acknowledge the support and guidance provided by my research supervisor throughout the project. Special thanks go to the students who carried out the measurements for this research. Finally, I extend my appreciation to those who reviewed this report and provided valuable feedback and suggestions.

References

1. Pischel, D., Buchbinder, J. H., Sundmacher, K., Lavrik, I. N. & Flassig, R. J. A guide to automated apoptosis detection: How to make sense of imaging flow cytometry data. *PLoS One* **13**, e0197208 (2018).

2. Luo, S. *et al.* Machine-Learning-Assisted Intelligent Imaging Flow Cytometry: A Review. *Advanced Intelligent Systems* **3**, 2100073 (2021).