**Artificial Intelligence-Driven Automation of Flow Cytometry Gating**

Gabriella Rivera

Applied Data Science Master’s Program

Shiley Marcos School of Engineering / University of San Diego [gabriellarivera@sandiego.edu](mailto:gabriellarivera@sandiego.edu)

John Vincent Deniega

Applied Data Science Master’s Program  
Shiley Marcos School of Engineering / University of San Diego   
 [jdeniega@sandiego.edu](mailto:jdeniega@sandiego.edu)

**ABSTRACT**

Flow cytometry is a biochemical process that measures the physical and chemical characteristic of cells in a liquid suspension. This method enables the identification and classification of various cellular populations, such as lymphocytes, monocytes, and granulocytes – from Peripheral Blood Mononuclear Cell (PBMC) samples. Clustering algorithms aim to objectively simplify the classification accuracy of a given population's cellular type to further immunological and clinical diagnostic purposes. By exploring algorithms like K-means, Agglomerative, and Gaussian Mixture Modeling, the dataset containing metrics on front and side scatter areas of cell scans, fluorescence on markers, and other dimensions can potentially lead to automatic gating and bring forth insights on cell population characteristics, resulting in increased analytical throughput and increased rate of medicinal and pharmacological breakthroughs.

1 Introduction

According to Brestoff and Frater (2022), flow cytometry is a cost-prohibitive cellular identification and pharmacological discovery process that both requires exceptional capital investment in infrastructure and is inherently difficult in adopting and implementing the latest techniques and technologies in the field. Open-source and low-cost options may serve as a viable bridge technology between capital-intensive investment cycles for laboratories to be able to continue evolving and increasing their analytical throughput without relying on the next hardware upgrade. By using existing mathematically-driven principles inherent to artificial intelligence, biochemists may be able to reduce the analytical inputs required to perform routine cellular classification and clustering of PBMCs. This labor-intensive process ultimately serves to evaluate the efficacy of experimental groups related to clinical trials that are required in the development of new life-saving medicines.

2 Background

Flow cytometry can be a capital-intensive process that requires significant investments in laboratory-grade biomedical equipment, dedicated graphics processing and tensor processing units, expansive random-access memory, and proprietary analytical software licenses. With Flow Cytometry Standard (FCS) files readily available on public repositories and by leveraging open-source and permissive license packages such as Scikit-Learn and Matplotlib to perform computational transformations to FCS data, we aim to discover cost-effective alternatives to expensive enterprise software licenses that perform flow cytometry analysis, which may result in significant reduction in the barriers to entry in biochemical flow cytometry.

Because cellular populations related to these FCS files number in the millions of records across multiple laboratory readings, this project will place heavy emphasis on dimensionality reduction in order to meet the constraints of being both cost-effective and hardware resource-efficient. Accomplishing such a feat would result in independent biochemical scientists to perform analyses without relying on exceptionally powerful computing hardware resources or costly proprietary enterprise-level software licenses.

2.1 Problem Identification and Motivation

As of this publication, flow cytometry gating is a manual process that involves a highly-trained biochemist to process and analyze the results of optical scans of cellular assays that may be further augmented by fluorescent substrates. Because of the complex and highly-dimensional nature of the data, these scientists rely on a best-practices approach based on their own respective processes and frameworks. Because of the potential variability of these processes and frameworks, the resulting findings from interpreting scan results is dependent on both the breadth and depth of methods of a given supervising scientist, thus resulting in both an increase in cost of analysis due to human error and omission as well as a reduction in consistency of results.

2.2 Definition of Objectives

The research team aims to utilize open-source and publicly-available resources from well-known algorithms known in data science to include principal component analysis, t-distributed stochastic neighbor embedding, and unsupervised clustering machine learning methods as well as FCS data hosted by FlowRepository (2020). Once data is cleaned for noise from scan data, the team aims to train models or machine-learning applications that have potential for value-added analysis relative to that of a typical human biochemist. Upon evaluation, success is generally defined when automated analysis reaches parity with a human analyst of at least 90% classification accuracy of PBMCs toward their respective dendritic cellular type on an unseen test set containing FCS scan data. In the event that this evaluation criterion is not met, further justification would have to be provided whether the measured degree of accuracy is acceptable relative to the speed of analyses.

3 Literature Review (related works)

Since 2016, a number of academic threads have been studied involving the advancement in flow cytometry, the iteration of methodologies when incorporating machine learning applications on FCS data, as well as different strategies in how to potentially automate the classification of cellular groups. By 2024, Ng et al. (2024) demonstrate maturity over an eight-year period that transitions the focus of academic research from the “what” normally seen in earlier works into the “how” with respect to interdisciplinary guidelines as well as quality control and assurance of future deployment of artificial intelligence in flow cytometry (p. 228).

3.1  FlowAI: Automatic and interactive anomaly discerning tools for flow cytometry data

FlowAI is a software package for the statistical computing language R, which Monaco et al. (2016) developed as a means to both clean FCS files from anomalies and to assess the resulting quality of the cleaned data normalized by the flow rate of a given reading. When flow rate abruptly changes during a scan, the readings may exhibit data inconsistencies. These data inconsistencies are considered anomalous and are discarded from the dataset. Using time-series analysis, the resulting dataset is broken into trend and cyclical components before being normalized by penalization function measuring absolute deviation of a data point from the median. Monaco et al. (2016) place an emphasis on data quality and anomaly handling, which are crucial considerations to flow cytometry, however they do not address the next step in automatic gating of cellular types, which is the focus of our research.

3.2  An open-source solution for advanced imaging flow cytometry data analysis using machine learning

Hennig et al. (2017) identify the challenges associated with the manual and subjective nature of flow cytometry, resulting in inconsistent in analysis. The given solution is to utilize open-source software, CellProfiler, to use raw image files to identify cell types from a flow cytometer image. Our research shares the open-source idea of being able to leverage existing machine learning algorithms to automatically classify these cell types. Contrasting the team of Hennig et al. (2017) on classification differs greatly in their use of visual image data as the basis for classification rather the numerical scan data from fluorescent biological marker excitation that is central to our approach (p. 202).

3.3  Comprehensive phenotyping of human dendritic cells and monocytes

Mair and Liechti (2020) identify the potential benefits in using biological markers to identify the phenotypes specific to dendritic cells and monocytes for cellular classification. This particular research focuses on a potentially more significant subset of biological markers and lineages that aim to more precisely identify different cellular categories as a result of their fluorescence excitation scan data. This work serves as the source data of our project which uses Python-based machine learning packages for automatic gating. A similar methodology was employed by Hennig et al. (2017) who instead synthesized with visual imagery data with the open-source software, CellProfiler.

3.4  Application of machine learning for cytometry data

Hu et al. (2022) acknowledge the complex challenge of highly-dimensional flow cytometry data and the potential for existing machine learning software packages to perform analysis on this type of data. This particular team first focuses on dimensionality reduction by means including Principal Component Analysis and stochastic methods, unsupervised and supervised machine learning methods to predict resulting clinical outcomes such as healthy populations versus diseased populations (p. 2). Our project aims to build on this research with greater training and tuning toward existing biological knowledge cross-validated across different FCS file scan results.

3.5  Recommendations for using artificial intelligence in clinical flow cytometry

Most recently, Ng et al. (2024) focuses on a more interdisciplinary approach to using artificial intelligence in flow cytometry with unique considerations for clinical risk management, quality control and assurance, and computational efficiency. This requires extensive consideration as to the narrative annotations required for clinical implementation. Though the article is comprehensive across multiple sectors related to flow cytometry and the technical and regulatory nuances required when applying artificial intelligence, it only provides general recommendations and guidance for future scientist who wish to leverage this new technology. Relative to our existing work, our research team aims to apply these general recommendations and implement them in an open-source and demonstrable product for flow cytometry automatic gating.

4 Methodology

Our platform approach is organized into several key subtopics, starting with data extraction. The FCS files were read using FlowCal, which were then transformed into NumPy and Pandas objects to facilitate compatibility with Interactive Development Environments (IDEs), specifically with Jupyter Notebook and Google Colab for our purposes. Exploratory Data Analysis (EDA) was then performed to generate data visualizations and detect outliers, which facilitated the data preprocessing. Dimensionality reduction measures were used to simplify the multi-channel complexity of the original data before feeding the sets into the various models and machine learning methods. Products for the final launch of the completed flow analysis product can be found at the following GitHub repository link at https://github.com/vanguardfox/ADS599.

4.1 Data Extraction and Data Structure Conversion

The flow cytometry dataset was acquired from FlowRepository, a public database for flow cytometry peer-reviewed experiments. It contains a staining panel from Mair and Leichti’s (2020) article that aimed to refine traditional and recently described markers for phenotyping dendritic cells and monocytes – cells that play critical roles in the immune system, and are thus indicators of immune response, disease status, and other markers of pharmacological efficacy. The panel data is composed of 23 fluorochrome markers, the time of collection, and forward scatter and side scatter measurements. There were 28 fluorescence channels in total, of which five of the channel wavelengths were unlabeled as a consequence of continuous data acquisition. About 2 million cells were collected per sample, which were reflected in the file sizes ranging between 267 to 405 megabytes for one PBMC FCS file. Additional compensation FCS files were also included in the dataset.

FCS data was parsed and ingested as a FlowCal.io.FCSData object, which is derived from a NumPy array. Data was found to be of float big-endian format, which was converted to little-endian format in a NumPy array in order to facilitate downstream visualization plots and other data transformations for analysis.

The available attributes from the FCS metadata were further parsed to retrieve the channel marker labels using the channel\_labels() method. The first three features for forward scatter area (FSC-A) and side scatter area measurements (SSC-A) were corrected and renamed to "FSC-A," "FSC-H," and "SSC-A," and the "Time" label was reiterated in the resulting NumPy array. This array was then converted into a Pandas DataFrame for better compatibility with further downstream visualizations. Finally, the formatted DataFrame was saved as a comma-separated values formatted file for computational compatibility purposes to be used for data preprocessing.

4.2 Data Feature Selection

Features were selected based on their relevancy to their ability to provide marker information on dendritic and monocyte cells, which were our target cellular populations. Although 23 fluorescence markers were used to identify specific cell surface proteins, a total of 28 channels were recorded when Mair and Leichti (2020) conducted the original data collection. The five unused channels with missing marker labels were discarded because they were blank, while the remaining 23 markers have known response ranges. To focus on the cellular pathways relevant for dendritic cell phenotyping (Figure 4.2.1), only those markers and their corresponding lineages relevant to dendritic cells were selected, with the remaining markers discarded as they have no value for our target cell population. Specifically, markers following the lineage through CD45RA, CD3, CD19, CD14, CD20, HLA-DR, CD123, CD11c, and Live Dead UV Blue were retained, along with Time and scattering measurements. This reduced the feature set to 13 with the rest of the lineages and subsequent markers pruned.

**Figure 4.2.1**

*Maecker et al. (2012) Dendritic Cell Lineage*

A diagram of cells and cells

Description automatically generated

4.3 Exploratory Data Analysis

Two-dimensional visualizations were plotted to identify general areas and priorities for cleaning the dataset. A SSC vs. Time scatter plot (see Figure 4.3.1) was created to identify inconsistencies during data acquisition as cells pass through the inflection point against the detection probe. This plot ensures that only cells collected during the stable portion of the sample run are included in the analysis. Figure 4.3.1 further illustrates the gating boundaries, which capture consistent readings across time and help exclude artifacts or outliers caused by fluctuations in the data acquisition process.

a

**Figure 4.3.1**

*Acquisition Plot*

A diagram of a graph

Description automatically generated with medium confidence

A histogram of the forward scatter area measurement (FSC-A) is shown in Figure 4.3.2. FSC-A is used to measure cell size in a given sample mixture. In this particular case, the resulting plot reveals three distinct peaks, which suggests at least three distinct cellular populations, corresponding to the expected cell types in PBMC as lymphocytes, monocytes, and granulocytes (left to right). The red gate is applied to exclude cellular debris, which typically appears as smaller events at the lower end of the FSC-A distribution.

**Figure 4.3.2**

*Cellular Debris Plot*

A blue graph with numbers and a red line

Description automatically generated

Figure 4.3.3 illustrates the removal of doublets or cell aggregates from the dataset, which are typically identified by an inconsistent ratio between FSC-A and FSC-H. Doublets tend to exhibit a higher FSC-H relative to FSC-A as they are larger due to the presence of two cells but still emit a "tall" scatter signal. In this case, the plot reveals a relatively small population of cell aggregates, identified by the gate on the y-axis. The x-axis limit is set further out to avoid truncating the monocyte population in the SSC-A vs. FSC-A plot, ensuring that all monocytes are still recalled while removing the as many doublets as possible.

**Figure 4.3.4**

*Single Cell Plot*

A diagram of a single cell

Description automatically generated

Finally, the Live/Dead™ UV Blue marker was used to exclude dead cells from the dataset. The dye binds to free amines present on the surface and interior of dead cells as a result of a broken cellular membrane, resulting in a higher fluorescent signal response than that of a live cell. In contrast, live cells emit a much weaker signal. As shown in Figures 4.3.5 and 4.3.6, live cells are gated to the left of the Live/Dead marker at 1000,000 relative fluorescence units (RFU) in both the original and log-transformed Live/Dead channels. This gating strategy cleaned/reduced the number of cell records to 1.96 million viable cells.

**Figure 4.3.5**

*Live Dead Cells: Linear Scale*

*A graph with a red line

Description automatically generated*

**Figure 4.3.6**

*Live Dead Cells: Logarithmic Scale*

A graph of a number of people

Description automatically generated with medium confidence

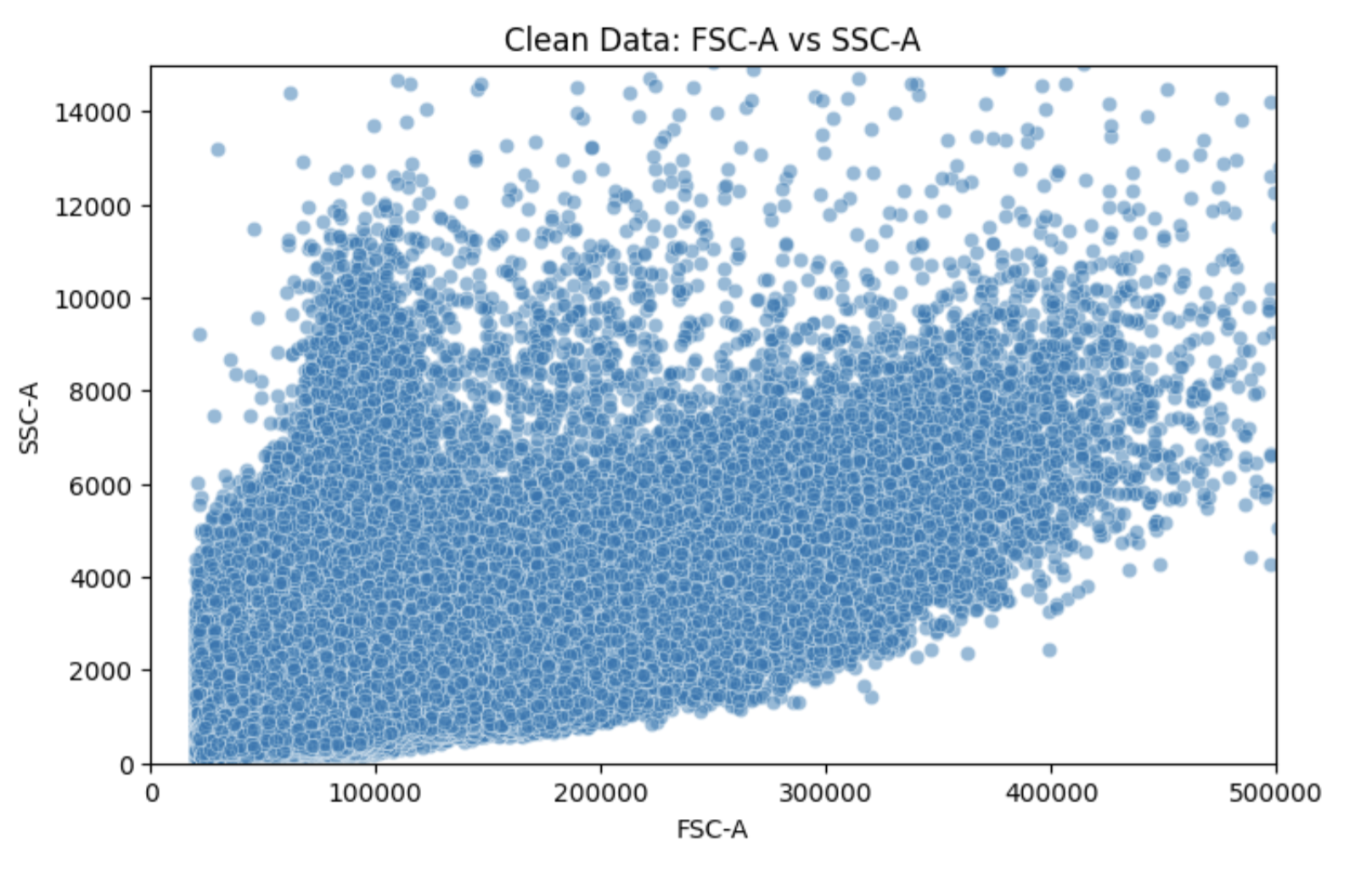
4.4 Data Preprocessing

According to FlowRepository (2020), the data that was obtained for this project was collected from four compensated donors. Because no further information is provided as to the selection criteria of these four donors and because the sample size of the millions of cells from the files was still constrained to these four donors, there may be bias in the data observed of unknown magnitude with respect to its representation of the overall global population. Additionally, upon inspection of the FCS file data, there is no personally-identifiable information present in the dataset that may in jeopardy of Health Insurance Portability and Accountability Act statutes that can be used identify a particular patient or donor. Flow cytometry only analyzes clinical data irrespective of the individual that provided the donation upon original data acquisition.

As alluded to during exploratory data analysis (EDA), data cleaning involved several steps to ensure the quality of the dataset before passing it through the model classification development. These steps included checking for missing values using the Klib package – with no missing values detected – removing irrelevant cell populations, identifying cellular debris, excluding doublets, and filtering out non-viable cells. Instances where the "Time" variable was outside the desired range were excluded by gating on values greater than 3 and less than 215. Additionally, cell populations with abnormal FSC and SSC characteristics were removed as those are indications that a cell reading is either debris or doublets. For FSC, cells were selected by gating for readings between 20,000 and 550,000 for FSC-A and for readings less than 200,000 for FSC-H. For SSC, a similar gating strategy was applied, where cells were retained only if SSC-A values were between 110 and 20,000. Finally, dead cells were excluded by applying a threshold on the Live/Dead UV Blue marker which keeps only those with values below 700 RFU. After these steps, the dataset was refined to include quality viable cells suitable for data splitting and model development. The cleaned dataset profile is shown in Figure 4.4.1.

**Figure 4.4.1**

*Clean Dataset Light Scatter Plot*



To prepare the data for model training and evaluation, the dataset was split into three distinct subsets: training, validation, and test sets. Initially, 20% of the data was reserved for testing using the train\_test\_split function, leaving the remaining 80% as the train\_val set. This initial test set is crucial because it ensures the model's final evaluation is done on data that has not been used during training or hyperparameter tuning, providing an unbiased estimate of its performance. The train\_val set was then further split, with 25% of the train\_val data allocated to the validation set, which is equivalent to 20% of the original dataset. This allocation is important for model tuning, as the validation set helps assess how well the model generalizes to new, unseen data while adjusting hyperparameters. The final result was a distribution of 60% for training, 20% for validation, and 20% for testing. These splits were chosen to ensure that there is enough data for training the model effectively, while maintaining a sufficiently sized validation set for model evaluation and tuning. The 20% test set is kept entirely separate to provide an independent performance assessment, minimizing the risk of overfitting and ensuring robust, realistic performance evaluation.

4.5 Dimension Reduction

To efficiently prepare the large datasets for modeling, we apply a series of data reduction techniques. First, downsampling is used to reduce the dataset size by selecting a representative subset using KMeans clustering and stratified sampling, ensuring proportional cluster representation. This step makes the dataset manageable for training while preserving key patterns. The next technique is Principal Component Analysis (PCA). It was applied to reduce dimensionality by transforming the data into a set of uncorrelated principal components which then highlights the most important features and minimizing noise. Finally, t-SNE was used for visualization, where t-SNE refines the data further by preserving local structures after the PCA global variance reduction. Downsampling, PCA, and t-SNE simplify the data, enhance model efficiency, and ensure that the most informative aspects are retained for accurate predictions.

4.5.1 Downsampling (Stratified Sampling)

Downsampling is applied to reduce the dataset size while preserving the structure and distribution of the data. In this case, KMeans clustering is first used to group the data into 10 clusters. Each cluster is formed by identifying patterns and similarities in the feature space, and the data points within each cluster are assigned a cluster label. Once the data is clustered, stratified sampling is performed to ensure that the downsampled data maintains the relative density of each cluster. Specifically, 10% of the samples are randomly selected from each cluster, with larger clusters being represented proportionally. This process ensures that no cluster is overrepresented or underrepresented in the downsampled data. The result is a smaller, more manageable dataset that retains the original data's structural properties, which we used to train the models effectively without losing key patterns in the data.

4.5.2 Principal Component Analysis (PCA)

After selecting features specific to dendritic cell markers and reducing the dataset to 12 columns, computationally expensive pairwise comparisons pose a challenge if the data is used directly. To address this hardware limitation, PCA is applied to reduce the data’s dimensionality. Using the elbow method to determine the optimal number of components, the cumulative explained variance plot (Figure 4.5.2.1) shows that PCA1 captures less than 95% of the variance, while the inclusion of PCA2 accounts for 97%. PCA3 is also included to enable 3D visualization of the 13 selected features (Figure 4.5.2.2), which provides an additional perspective on the data's structure and relationships, helping to better distinguish patterns that may not be as apparent in lower-dimensional representations. By transforming the 13 columns into three principal components, the variance is maximized while simplifying the data which then improves the efficiency of downstream clustering and modeling steps.

**Figure 4.5.2.1**

*PCA Cumulative Explained Variance Plot*

A graph of a number of components

Description automatically generated

**Figure 4.5.2.2**

*3D PCA Plot of Training Dataset*

A graph of different colors

Description automatically generated with medium confidence

4.5.3 T-distributed Neighbor Embedding (T-SNE)

Using t-SNE helps analysts capture local structures in high-dimensional data. Using three resulting components from PCA, t-SNE aids in being able to visualize different clusters while still being memory-efficient, which is a key consideration for our cost-effective solution. Because about 99% of the variance is still captured with only three components, the loss of one-percent of the data for significant memory efficiency directly addresses expected hardware limitations with a computationally intensive algorithm such as t-SNE. Further, t-SNE directly addresses the subjectivity issue that lends to analysts potentially being inconsistent across multiple scatter plots. As such, this method provides clearer and more objective population boundaries for the purposes of gating where different clusters may be isolated for further downstream analysis as shown in Figure 4.5.3.1.

**Figure 4.5.3.1**

*Working t-SNE Gating of CD19 vs. CD3 Markers*

A diagram of a cluster of dots

Description automatically generated with medium confidence

4.6 Modeling

To identify the best-performing classification model for the dataset, we tested three preprocessing techniques with three clustering algorithms: Gaussian Mixture Models (GMM), DBSCAN, and KMeans. These algorithms were chosen for their ability to identify different types of cell population clusters in flow cytometry data. GMM is effective at detecting overlapping or complexly shaped clusters, which are common in biological datasets. DBSCAN can find clusters of arbitrary shape and is robust to outliers, making it useful for noisy data. KMeans is best suited for well-separated, roughly spherical clusters, which are often found in distinct cell populations. To apply these techniques, we first downsampled the dataset and ran each clustering algorithm. We repeated this process for the PCA-reduced dataset and the PCA with t-SNE dataset to evaluate how dimensionality reduction affected cluster identification. By systematically testing these preprocessing steps, we compared the algorithms based on processing time and classification accuracy. This approach allowed us to select the best model for classifying flow cytometry data.

4.6.1 GMM

4.6.2 DBSCAN

4.6.3 K-means Clustering

**References**

Beckman Coulter. (2022). *Automatic gating - Beckman Coulter*. Flow Cytometry. <https://www.beckman.com/flow-cytometry/software/cytobank-premium/learning-center/automatic-gating>

Brestoff, J. R., & Frater, J. L. (2022). Contemporary challenges in clinical flow cytometry: Small samples, big data, little time. *Journal of Applied Laboratory Medicine*, *7*(22), 931-944. <https://doi.org/10.1093/jalm/jfab176>

Cossarizza, A., Chang, H. D., Radbruch, A., Acs, A., Adam, D., Adam-Klages, S., Agace, W. W., Aghaeepour, N., Akdis, M., Allez, M., Almeida, L. N., Alvisi, G., Anderson, G., Andrä, I., Annunziato, F., Anselmo, A., Bacher, P., Baldari, C. T., Bari, S., Barnaba, V., … Zychlinsky, A. (2019). Guidelines for the use of flow cytometry and cell sorting in immunological studies (second edition). *European journal of immunology*, *49*(10), 1457–1973. <https://doi.org/10.1002/eji.201970107>

FlowRepository. (2020). FlowRepository ID FR-FCM-Z32U. Retrieved from <http://flowrepository.org/experiments/3166/download_ziped_files>

Hennig, H., Rees, P., Blasi, T., Kamentsky, L., Hung, J., Dao, D., Carpenter, A. E., & Filby, A. (2016). An open-source solution for advanced imaging flow cytometry data analysis using machine learning. *Methods*, 112(2017), 201–210. <http://dx.doi.org/10.1016/j.ymeth.2016.08.018>

Hu, Z., Bhattacharya, S., & Butte, A. J. (2022). Application of machine learning for cytometry data. *Frontiers in immunology,* 12, 787574. <https://doi.org/10.3389/fimmu.2021.787574>

Lee, J. A., Spidlen, J., Boyce, K., Cai, J., Crosbie, N., Dalphin, M., Furlong, J., Gasparetto, M., Goldberg, M., Goralczyk, E. M., Hyun, B., Jansen, K., Kollmann, T., Kong, M., Leif, R., McWeeney, S., Moloshok, T. D., Moore, W., Nolan, G., Nolan, J., … Brinkman, R. R. (2008). MIFlowCyt: the minimum information about a Flow Cytometry Experiment. *Cytometry. Part A: the journal of the International Society for Analytical Cytology*, 73(10), 926–930. <https://doi.org/10.1002/cyto.a.20623>

Maecker, H. T., McCoy, J. P., & Nussenblatt, R. (2012). Standardizing immunophenotyping for the Human Immunology Project. *Nature Reviews Immunology,* 12(3), 191–200. <https://doi.org/10.1038/nri3158>

Mair, F., & Leichti, T. (2020). Comprehensive Phenotyping of Human Dendritic Cells and Monocytes. *Journal of Quantitative Cell Science*, 99(3), 231–242. <https://doi.org/10.1002/cyto.a.24269>

Monaco, G., Chen, H., Poidinger, M., Chen, J., Magalhaes, J., & Larbi, A. (2016). FlowAI: Automatic and interactive anomaly discerning tools for flow cytometry data. *Bioinformatics*, 32(16), 2473–2480. <https://doi.org/10.1093/bioinformatics/btw191>

Ng, D. P., Simonson, P. D., Tarnok, A., Lucas, F., Kern, W., Rolf, N., Bogdanoski, G., Green, C., Brinkman, R. R., & Czechowska, K. (2024). Recommendations for using artificial intelligence in clinical flow cytometry. *Cytometry part b: Clinical cytometry*, 106(4), 228–238. <https://doi.org/10.1002/cyto.b.22166>

Spidlen J, Breuer K, Rosenberg C, Kotecha N and Brinkman RR. (2012). FlowRepository - A Resource of Annotated Flow Cytometry Datasets Associated with Peer-reviewed Publications. *Cytometry A*. 81(9), 727-31. <https://doi.org/10.1002/cyto.a.22106>