Predicting Molecular Phenotypes with Single Cell RNA Sequencing Data: An Assessment of Unsupervised Machine Learning Models

Anastasia Dunca
Massachusetts Institute of Technology, Cambridge, MA 02139, e-mail: adunca08@mit.edu

Project Advisor: Dr. Frederick R. Adler
Department of Mathematics, University of Utah, UT 84112, e-mail: adler@math.utah.edu

Abstract

According to the National Cancer Institute, there were 9.5 million cancer-related deaths in 2018. A challenge in improving treatment is resistance in genetically unstable cells. The purpose of this study is to evaluate unsupervised machine learning on classifying treatment-resistant phenotypes in heterogeneous tumors. This is done with analysis of single cell RNA sequencing (scRNAseq) data using a pipeline and evaluation metrics. scRNAseq quantifies mRNA in cells and characterizes cell phenotypes. One scRNAseq dataset is used (tumor/non-tumor cells of different molecular subtypes and patient identifications). The pipeline consists of data filtering, dimensionality reduction with Principal Component Analysis, projection with Uniform Manifold Approximation and Projection, clustering, and evaluation. Nine approaches for clustering (Ward, BIRCH, Gaussian Mixture Model, DBSCAN, Spectral Clustering, Affinity Propagation, Agglomerative Clustering, Mean Shift, and K-Means) are evaluated. Seven models divided tumor versus non-tumor cells and molecular subtype while six models classified different patient identification (13 of which were presented in the dataset); K-Means, Ward, and BIRCH often ranked highest with $\sim 80\%$ accuracy on the tumor versus non-tumor task and $\sim 60\%$ for molecular subtype and patient ID. An optimized classification pipeline using K-Means, Ward, and BIRCH models was evaluated to be most effective for further analysis. In clinical research where there is currently no standard protocol for scRNAseq analysis, clusters generated from this optimized pipeline can be used to understand cancer cell behavior and malignant growth, directly affecting the success of treatment.

Keywords: Unsupervised machine learning, single cell RNA sequencing, gene expression, clustering, tumor heterogeneity, dimensionality reduction, molecular subtype, patient identification, classification

1 Introduction

1.1 Tumor Heterogeneity

Tumor heterogeneity is a state in which cancer cells within the same tumor are functionally different from each other [20]. As a patient's cancer progresses, homogeneous cell subpopulations within the heterogeneous tumor cell population–genetically distinct groups of tumor cells

descending from a common ancestor [24]–can be identified and their evolutionary trajectory can be followed with response and resistance to treatment. This hypothesis regarding tumor heterogeneity was first made in 1976 with Peter Nowell who suggested that tumors have multiple heterogeneous subclones [20]. Individualized cancer treatment was developed in order to stop treating cancer as if it was homogeneous. In particular, tumor heterogeneity can be associated with resistance to cancer treatment. Because the subclonal populations within a tumor can be sensitive or resistant, generalized cancer treatment will become repeatedly less effective with more rounds of treatment. The treatment will only be effective for the portion of the tumor that is not yet resistant. By identifying subclonal populations, targeted therapy may be developed to specifically target the oncogenic causes [26].

1.2 Single Cell RNA Sequencing

Single cell RNA sequencing (scRNAseq) has dominated bioinformatics research with the invaluable information it provides such as complex and rare cell populations, regulatory relationships between genes, and the trajectories of distinct cell lineages in development [16]. Different cells may express different genes at different levels. These differences are a key to understanding the behavior of cells. The absolute quantity of RNA molecules is very small in a single cell, so experiments often consist of billions of cells. When RNA is extracted from a group of cells, all information about the difference between them is lost, so the data are a mean expression of each gene. To link gene expression data to individuals cells, RNA from each cell is uniquely barcoded and sequenced. For each sequenced molecule, the information attained is the ID of the cell–called the barcode–and its own unique ID–called the unique molecular identifier. The resulting data from the sequencing is formatted in a genes-by-cells table P, where P[i, j] is an expression of gene i in cell j [19].

1.3 Machine Learning

Machine learning is a modern computational tool that is at the intersection of linear algebra, statistics, and computer science. Machines automatically learn and improve from experience without being explicitly programmed to produce certain outputs; essentially, they make predictions. In the large umbrella of machine learning, there are many approaches that stem from the two types of learning: supervised and unsupervised learning. *Supervised learning* is primarily conducted when the data have labels (a target variable on a given set of predictors). *Unsupervised learning* is primarily conducted when the data lack labels. Given the unlabeled nature of scRNAseq data as well as the lack of independent variables, we here use unsupervised learning. The dataset that is clustered lacks labels, and the clusters are then compared to the truth data that does have labels.

In our work, unsupervised learning is done with cluster analysis and dimensionality reduction. *Dimensionality reduction* reduces the number of variables (dimensions) in the data while preserving often 80% of the variance. *Clustering* divides the data into groups based on patterns within the data (here, gene expression). In optimal clusters, all data points should be close to each other and the groups should be separated. With scRNAseq data, biologists use clustering to identify how many different cell states there are.

In this work, nine different models are evaluated in the five categories of clustering: hierarchical, density based, model based, centroid based, and graph based clustering. The hierarchical clustering methods tested are Agglomerative (average linkage) Clustering, Ward Clustering, and BIRCH Clustering (Balanced Iterative Reducing and Clustering using Hierarchies). The density-based clustering methods tested are DBSCAN (Density Based Clustering of Spatial Applications with Noise). The model based clustering is tested with the Gaussian Mixture Model. The centroid

based clustering is tested with K-Means and Mean Shift Clustering. Graph-based clustering is tested through Affinity Propagation and Spectral Clustering.

1.3.1 Unsupervised Learning Models

Here, we will present the underlying mathematical methods of each of the unsupervised learning models used in this work (BIRCH, Ward, Spectral Clustering, Gaussian Mixture Model, DBSCAN, Affinity Propagation, Agglomerative Clustering, Mean Shift, K-Means).

BIRCH clusters multi-dimensional metric data points to produce the most optimal clusters using dynamic and iterative methods. A primary quality of BIRCH clustering is that it can create a decent structure within one scan of the data, and improve with a few additional scans. It can also handle outlier data points effectively. The BIRCH model has been proven to be suitable for large databases [10]. A Clustering Feature (CF) vector is defined as an ordered triple (N, LS, SS). 'N' represents the number of data points in the cluster, 'LS' is the linear sum of the data points and 'SS' is the squared sum of the data points in the cluster. A cluster is a set of data points, but the CF vector is stored as a summary. In this case, each data point would be a cell. The groups of cells would be clusters (or in this case CF vectors)–grouped together by their gene expression information. A CF vector is sufficient for calculating all the measurements needed for making clustering decisions in BIRCH, and makes calculations a lot faster. The CF vectors go into a CF tree, which is made up of leaf and non-leaf nodes. Nodes essentially represent a group of observations. A non-leaf node contains multiple entries of the form $[CF_i, child_i]$, and "$child_i$" is a pointer to the parent node's $i-th$ child node, and CF_i is the CF of the sub-cluster represented by this child. A leaf node contains multiple entries, each entry made up of a CF vector. Both of these types of nodes represent a cluster made up of all the sub clusters represented by the node's entries. A CF tree will be built dynamically as data objects are inserted as the algorithm continues its scan of the dataset, and is used to guide each new insertion into the correct sub cluster for clustering purposes. A CF tree is a very compact representation of the dataset because each entry in a leaf node is not a single data point but a subcluster. Different distance metrics can be used to insert a new entry into a CF tree by determining the closest child node for a leaf to "absorb." To summarize, BIRCH clustering works by first loading all the data into memory by building a CF tree composed of CF vectors. From there, it creates a smaller CF tree by scanning the leaf entries in the initial CF tree to rebuild a smaller CF tree while removing more outliers and grouping crowded sub clusters into larger ones. Then it performs global clustering by clustering all leaf entries. Lastly, it refines the clusters [28].

Ward looks at cluster analysis as an analysis of variance problem [2]. Ward's approach is *agglomerative*, that is, it begins at the smallest individual structure and then progresses to a global structure.

The method starts with each point in a cluster by itself, where the sum of squares of Euclidean distance is 0. This is due to the fact that pairwise distances between points in the clusters is zero since the same cluster is compared to itself. Then, it merges two clusters, in order to produce the smallest increase in the sum of squares (by finding the smallest merging cost, seen below). The method keeps merging clusters until k clusters is reached. With each cell being a data point, the merging cost would be calculated by the differences in gene expression between each cell. The merging cost is the increase in sum of squares when merging two clusters.
Given N (N cell lines) d-dimensional (d genes) data points in a cluster $\{\vec{X_i}\}$ where $i = 1, 2, ..., N$,

center of cluster $\{\vec{X_0}\}$ two clusters A and B, define the merging cost [5]:

$$\Delta(A,B) = \sum_{i \in A \bigcup B} \|\vec{X_i} - \vec{X}_{0A \bigcup B}\|^2 - \sum_{i \in A} \|\vec{X_i} - \vec{X_{0A}}\|^2 - \sum_{i \in B} \|\vec{X_i} - \vec{X_{0B}}\|^2 = \frac{N_A \cdot N_B}{N_A + N_B} \|\vec{X_{0A}} - \vec{X_{0B}}\|^2 \tag{1}$$

Spectral Clustering is a technique rooted in graph theory. It identifies communities of nodes in a graph, based on the edges connecting them. It uses information from the eigenvalues of special matrices built from the dataset [14]. It first computes a similarity graph [12]. Then, using the similarity graph, it projects the data onto a low-dimensional space. Data points in the same cluster may be far away, even farther away than points in different clusters. The space is transformed so that when two points are close, they are always in the same cluster, and when they are far apart, they are in different clusters. After this, the clusters are created using Graph Laplacians, eigenvalues, and eigenvectors [12]. Depending on the values of these quantities, nodes with values higher than a certain threshold are put into clusters, and nodes lower than the threshold are put into others.

Gaussian Mixture Model is a probabilistic model for representing normally distributed subpopulations within an overall population. Essentially, it assumes all data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters[3]. The Gaussian–or normal–distribution is the most commonly used distribution in modeling real world unimodal data[23]. The Gaussian mixture model implements the expectation-maximization algorithm, which tries to use existing data to determine the optimum values for these variables and then finds the model parameters for the Gaussian Mixture Model[23]. This consists of two steps:

- E-step: the available data is used to estimate the values of the missing variables
- M-step: the complete data is used to update the parameters

The algorithm goes back and forth between these two steps, constantly generating new probabilities (the chance that one data point belongs to a certain cluster) for each data point until the number of mixture components (or clusters) is reached[23].

DBSCAN is Density-Based Spatial Clustering of Applications with Noise. The algorithm focuses on point density in which points close in distance are grouped together whereas outliers are found in regions of low point density [15]. It starts with an arbitrary cell/data point. It extracts the neighborhood of the point using the value of ϵ where all points are under distance ϵ from the neighborhood. If the neighborhood achieves the MinPts threshold then the clustering process starts and the point is marked as a core point and a cluster formation starts; if it does not reach that threshold, then the point is labeled as noise (a point that is marked as noise may be revisited and be part of a cluster). If a cluster formation is started, then the points part of its ϵ neighborhood become part of the cluster [27]. If all of these points are also marked as core points (having an ϵ neighborhood reaching the threshold), the points part of those neighborhoods are also added to the cluster. After that cluster is fully formed, the algorithm randomly chooses another observation among the points that have not been visited in previous steps, and the same procedure is performed. This process continues until all points have been visited [7].

Affinity Propagation is a graph based clustering algorithm that finds *exemplars* which are members of the input dataset that are representative of clusters. It takes a similarity matrix as input and identifies exemplars based on certain criteria such as availability and responsibility. The algorithm proceeds by alternating between two message passing steps iteratively to update the responsibility and availability matrices. Specifically, responsibilities are sent from data points to candidate exemplars and indicate how strongly each data point favors the candidate exemplar over other candidate exemplars; availabilities are sent from candidate exemplars to data points and indicate to what degree each candidate exemplar is available as a cluster center for the data point [13]. With the intake of a dataset of cells, Affinity Propagation calculates a similarity matrix, responsibility matrix, availability matrix, and criterion matrix. The criterion matrix is created once both responsibility and availability matrices are done updating; it is the sum of the responsibility and availability matrices. The element with the highest criterion value in each row of the dataset would be an exemplar. Elements corresponding to the rows which share the same exemplar are clustered together [25].

Agglomerative Clustering Average Linkage is a type of hierarchical clustering where each observation starts as its own cluster [6]. The first step merges together the two closest clusters into one cluster. Then, the next closest clusters are grouped together and this process is repeated until all observations are in a single cluster. The output of Agglomerative Clustering is a dendrogram showing the hierarchy of relationships found within the dataset. Agglomerative Clustering itself has different approaches that distinguish themselves based on distance metrics. In this paper, average linkage is used.

In average linkage clustering, the distance between two clusters is defined as the average distance between all pairs of objects where each pair is made up of one object from each group. The distance D for two clusters r and s is computed as:

$$D(r, s) = T_{rs}/(N_r \cdot N_s) \tag{2}$$

where T_{rs} is the sum of all pairwise distance between cluster r and cluster s and N_r and N_s are the sizes of the clusters r and s respectively. At each stage of hierarchical clustering, the clusters r and s for which D is the minimum are merged.

Mean Shift is a centroid-based algorithm which uses the kernel density estimation to assign each point to a cluster. It works by placing a kernel–or weighting function–on each point (cell) in the dataset. Adding all of the kernels up generates a probability surface or density function. The mean shift model exploits the kernel density estimation by predicting the behavior of the points if they were all at the peak on the KDE surface by shifting each point "uphill" until it reaches a peak. The peak is the centroid of that cluster. Once it reaches that peak, it is assigned to that cluster. It starts by making a copy of the original dataset and freezes the original points. The copied points are shifted against the original points. Each point is moved closer to the nearest KDE surface peak in each iteration [21]. The shift function is called iteratively for each point until the point is not shifted by much distance any longer–it converges on a peak. The algorithm stops when each data point has converged on a peak (again, each peak is the centroid of a cluster)[8]. Based on the peaks that each data point converged on, each data point is assigned to the cluster where the peak is the centroid of that cluster.

K-Means finds a user-specified number of clusters (k) which are represented by their centroids. It starts out by choosing k initial centroids, which are arbitrary observations (cells). Each point

is then assigned to the closest centroid and each collection of points assigned to a centroid is a cluster. The centroid of each cluster is then updated based on the points assigned to the cluster. The assignment and update steps are iterated until no points change clusters or the centroids remain the same. To measure the quality of clustering in each iteration, the sum of the squared error (SSE) is used. The error of each data point is calculated which is also its Euclidean distance to the closest centroid $\vec{X}_0$ of the cluster, and then the total sum of the squared errors is computed. A smaller squared error means the centroid is a satisfactory representation of the points within the cluster [22]:

$$SSE = \sum_{j=1}^{k} \sum_{i=1}^{N} ||\vec{X}_i^{(j)} - \vec{X}_0^{(j)}||^2 \tag{3}$$

1.4 Comparison of Models Table

Category	Model(s)	Key Differences Between Models
Hierarchical	BIRCH, Ward, Agglomerative	Metrics for similarity
Graph Based	Spectral Clustering, Affinity Propagation	Metrics for distance and connectivity
Density Based	DBSCAN	
Model Based	Gaussian Mixture Model	
Centroid Based	K-Means, Mean Shift	Quality of clusters

Table 1: Models Evaluated

1.5 Contributions of this Work

Single cell RNA sequencing (scRNAseq) is an emerging technology that requires modern, efficient methods of data analysis to properly handle big data and produce useful results for computational research. Currently, there is no standard scRNAseq analysis pipeline, nor has any thorough study been done on which clustering models respond best to scRNAseq data. In this study, we investigate which unsupervised learning methods are most effective and efficient for scRNAseq data analysis through rigorous evaluation. After experimentation, it was seen that for multiple tasks the highest performing models were BIRCH, K-Means, and Ward. With these models, an optimized pipeline was constructed that can be used for future data analysis. With knowledge of the highest performing models, it is also possible to combine the underlying methodology to create an even more effective model that can respond best to scRNAseq data.

The paper is organized as follows. In Section 2, we discuss our methodology for analysis and evaluation including brief backgrounds for each computational tool and technique. In Section 3, our findings are presented with visualizations of clustering accuracy, analysis over 100 runs, and a brief discussion of randomization tests conducted to validate the evaluation method. Section 4 discusses the conclusions that can be drawn from our work.

2 Methodology

In this section, we present the materials and methods used to analyze the scRNAseq. We will also discuss the rationale behind the methods along with more mathematical principles in the tools used.

2.1 Pipeline

The goal for analysis in this work is to separate the entire group of cells represented by the dataset into specific subgroups distinguished by: tumor v. nonTumor, patient ID (or non-identifiable patient label), and molecular subtype. To elaborate, tumor v. nonTumor separates cells based on if they are tumorous or not tumorous; non-identifiable patient label separates cells based on which patient they came from; and, molecular subtype distinguishes cells on their different molecular compositions. Each of the clustering visualizations created by the clustering models is evaluated on accuracy by comparing true state to predicted state. In our study, due to the stochastic nature of the models, the pipeline is run 100 times repeatedly, producing clusters each time. For each run of the pipeline, the accuracy of the clusters is measured, recorded, and then the 100 calculated accuracies are averaged to produce an accuracy score for each clustering model. Then, the actual true states are projected onto the cluster to examine the true state of each data point and the general structure of the clusters (Figures 6, 7, 8).

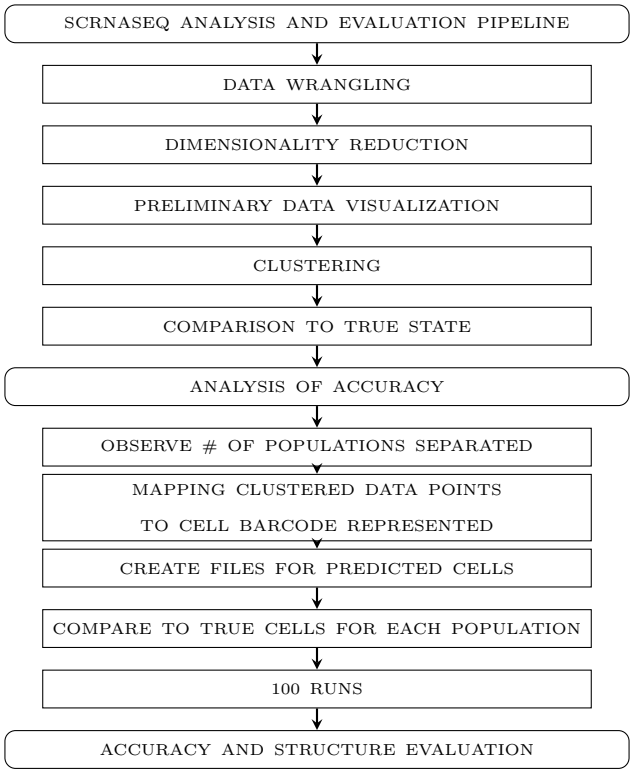


Figure 1: Observation Flowchart

2.2 Data

The source of the data was the City of Hope. Sequencing data from tissues obtained from an autopsy of eleven patients performed on February 8, 2019 was used in this project. The data was shared from my advisor's collaborator, Dr. Jeffrey Chang.

The data used in this study is a gene expression matrix with gene counts per million cells. The initial shape of the dataset was 57822 genes by 518 cells; after filtering outliers, the shape analyzed was 6563 genes by 518 cells. In terms of data, the dataset had 6563 features and 518 observational units. For the singular pipeline in development, three goals are assigned: successful separation by

cell state (tumor versus nonTumor), molecular subtype (estrogen receptor positive, double positive, triple negative breast cancer, and human epidermal growth factor), and non-identifiable patient labels (BC01 (26 cells), BC02 (56 cells), BC03 (37 cells), BC03_LN (55 cells), BC04 (59 cells), BC05 (77 cells), BC06 (25 cells), BC07 (51 cells), BC07_LN (53 cells), BC08 (23 cells), BC09 (29 cells), BC10 (16 cells), BC11 (11 cells)). These data were sequenced from the cell lines of estrogen receptor positive and negative cells.

2.3 Materials

The computational tools primarily used in this research were Python 3, JupyterNotebooks, and Anaconda. The libraries used were sklearn, numpy, PCA, matplot, umap, pandas, and scipy.

2.4 Exploratory Data Analysis and Data Filtering

The necessary libraries and machine learning tools need to be imported before any algorithms can be made. The scRNAseq dataset is loaded as well as the features (genes) and barcodes (cells) labels are assigned. For filtering and EDA (exploratory data analysis), it is important to understand the dimensions of the data as well as have a good image of the dataset. Plotting the current unfiltered data will allow for filtering opportunities and observations of total noise. In this case, *UMI's* (Unique Molecular Identifiers) per cell and per gene will be identified. This will be used to choose high and low expressed genes to filter out. The purpose of identifying and eliminating the outliers is so that as the machine learning algorithms learn, the outlier values will not have a heavy impact of the decisions that the model makes. Technical errors such as drop out events–when a gene is observed at low or moderate expression in one cell but no expression in another cell of the same type–it is important to clean the data for optimal performance of the models.

Based on the EDA, outliers are genes with an expression of below 100 and above 1,000,000. Only extreme outliers should be filtered out. Then, the coefficient of variation is plotted to define the highly variable genes in the dataset. Filtration is performed twice more by keeping only the highly variable genes and removing cells with very low expression.

2.5 Dimensionality Reduction with Principal Component Analysis

A *Scree Plot* is a diagnostic tool to check whether PCA works well on the current data. Principal components are made in the order of the amount of variation they cover: the first component captures the most and it decreases from there [11]. In PCA, there are as many principal components as features. The scree plot should be an indicator of what principal components to keep; ideally, the graph should somewhat look like an elbow, and the start of the elbow is the cut off (or the maximum number of principal components to include) [4]. The y axis is eigenvalues which

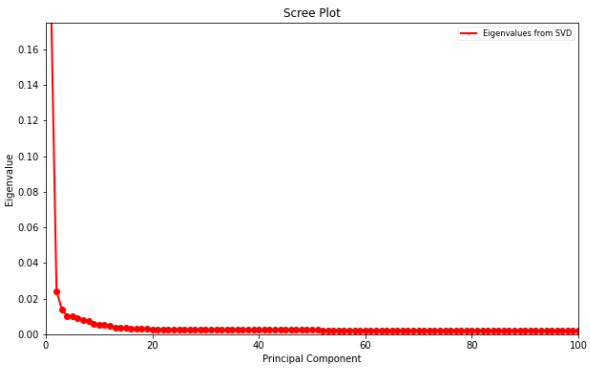


Figure 2: Scree Plot

means the amount of variation kept with each component. In this project, the kept principal components should be able to represent at least 80% of the total variance in the dataset. This allows for the majority of the variance to be kept, preserving structure, while quickening the speed of analysis as opposed to keeping 90-95%.

The total number of features in the dataset is approximately 6000. Minor skews can be observed around 50-100 components.

PCA is a dimensionality reduction technique which performs feature extraction on a dataset. [11]. PCA works by calculating a matrix that summarizes how each of the variables relate to one another. This matrix can be broken up into two components: direction and magnitude. Then, the data are aligned with the important directions. With direction selection then data compression, the dimensionality of the feature space is reduced; however, the data are only transformed into different directions, so all original variables are still in the model. PCA is a preprocessing step that keeps most variance in the data with little information loss, but greatly speeds up time efficiency in later steps such as UMAP and clustering. After testing out different components to keep, ranging from 5 to 100, we ended up keeping 100 principal components for all models (reducing the dataset from about 6000 to 100 dimensions) since that preserved structure the best with a decent runtime. PCA was performed after the data was filtered and before preliminary visualization with UMAP.

2.6 Preliminary Visualization with UMAP

The reason for using PCA before UMAP was for an additional preprocessing step that could preserve most variance within the dataset while keep runtime reasonable. After reducing the dimensions with PCA, *UMAP* reduces the dimensions to two and projects the data onto a 2D plane, to help with preliminary visualization of the clusters[9]. One large benefit of this step is that it will help visualize the difference in performance between clustering techniques. UMAP 1 versus UMAP 2 is plotted in the figure below [19].

UMAP constructs a high dimensional graph representation of the data then optimizes a low-dimensional graph to be as structurally similar as possible [9]. UMAP first makes a high dimensional graph, and then makes the graph "fuzzy" by decreasing the likelihood of connection for each data point as the radius of each neighborhood of points grows. By stipulation that each point must be connected to at least its closest neighbor, UMAP preserves local structure in balance with global structure. The image shown is the data without any clustering performed yet; this is a plot of the points without having been sectioned into groups.

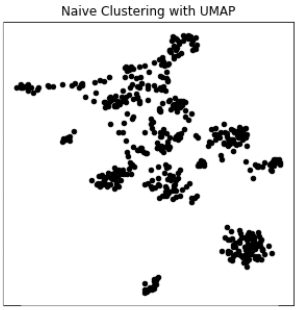


Figure 3: Preliminary Clustering

2.7 Parameter Tuning

After principal components have been selected with PCA and the embedding was projected with UMAP, multiple clustering algorithms can be tested for data analysis and for a consistency eval-

uation. When implementing clustering algorithms in Python, specific parameters (or these can be called arguments to the function) that dictate the visual aspect of the clusters (separation between clusters, number of clusters, minimum distance between points to be considered a neighborhood, etc.) need to be adjusted. These parameters were adjusted through trial and error, gauging for consistency amongst clustering methods within partitions and cluster sizes. Tuning took place until a point of "stable" clustering was reached–a consistency among different techniques was found and the clusters didn't drastically change between hyperparameter values[1]. Nine clustering algorithms will be tested: Affinity Propagation, Mean Shift, Spectral Clustering, Ward, Agglomerative Clustering, DBSCAN, BIRCH, K-Means, and the Gaussian Mixture Model.

The first step is to define all of these parameters into a dictionary. These were: *quantile* (where a sample is divided into adjacent subgroups), *epochs* (one complete presentation of the dataset to be learned by a learning machine), *damping* (measure to describe how rapidly oscillations decay from one investigation of a data point to the next), *preference* (task of learning to predict an order relation on collection), *n_neighbors* (how many points are assumed to be connected to one point), and *n_clusters* (the amount of clusters differentiated). These parameters are optimized with fine tuning and iterative examination.

2.8 Comparison

Below are three diagrams that show the cluster analysis of the nine models. The difference between each figure is the number of clusters created from the model. Specifically, clustering based on molecular subtype differs from clustering based on tumor ID because molecular subtype has four different subgroups, and tumor ID has only two. The clustering algorithms did not use any labels when clustering. When using the phrase "based on differences in [subgroup type]," this refers to what each visualization represents, not the method of clustering. Each different color shows a different cluster. From these, we can discuss an aesthetic analysis of how each model performed based on the look of the clusters themselves.

The first task is to distinguish the cells based on differences in molecular subtype. With four different molecular subtypes within the group of cells, the clustering algorithm should separate the cells into four separate subgroups. Figure 4 shows this task, highlighting that Affinity Propagation, Agglomerative Clustering, Mean Shift, Spectral Clustering, and DBSCAN have large differences from the rest of the models. Affinity Propagation differed in the number of clusters created (seen with the colors) as well as the partitions, where it divided up the large group of cells in the center of the plot into four subgroups and further grouped the cells separated by space. Mean Shift, Agglomerative, and DBSCAN Clustering divided the cells into 2, 3, and 4 groups, respectively. All of them kept the large group of cells in the center of the plot together while only distinguishing another group in the bottom right of the plot. Spectral Clustering also split the group of cells into four groups; however, instead of keeping the large group of cells in one group, it split that group into two and kept the small group of cells in the bottom right corner with the other large group of cells (by coloring it orange like the other group). The rest of the models (Ward, BIRCH, Gaussian Mixture Model, and K-Means) have similar structure with the same number of clusters and roughly the same partitions. All four of these split the group of cells into four groups by splitting the large group of cells in the center of the plot into approximately three groups with the group of cells in the lower right corner as the fourth group. All of these partitions can be seen with the colors that the clustering models assigned to each plotted point.

With the same dataset, the data can also be separated according to cell state (tumor versus nonTumor) which results in two subpopulations. In Figure 4, the models attempted to separate the data into groups based on molecular subtype whereas in Figure 5, the models attempted to

separate the data into groups based on cell state. This meant that the models separated the data into different numbers of groups, and the evaluation of accuracy (conducted later) was based on different labels. For tumor cells, Figure 5 shows that Affinity Propagation and DBSCAN are extremely different from the rest of the models, for they identify a substantially larger number of clusters. BIRCH and K-Means have roughy the same partitions while the rest of the models (Spectral Clustering, Ward, Agglomerative, Gaussian Mixture Model, and Mean Shift) also have the same partitions. BIRCH and K-Means both split the large group of cells in the approximate middle of the plot, using distance to divide the groups. Spectral Clustering, Ward, Agglomerative, Gaussian Mixture Model, and Mean Shift keep the large group of cells in one cluster and create a second cluster from the group of cells outside of the large group in the bottom right group of cells in the plot. Every model except Affinity Propagation and DBSCAN created two clusters. Affinity Propagation split the large group of cells into many smaller groups as indicated by the colors. DBSCAN performed similarly to Affinity Propagation with many subgroups, splitting many groups by tiny spaces found within the large cluster. Because there were only two groups to cluster, the consistency of partitioning for the clustering algorithms was higher than with molecular subtype and patient ID.

The third goal is to separate the entire dataset into thirteen distinct clusters representing different non-identifiable patient labels (patient ID). Although we have 11 different patients, two extra groups are included to represent the lymph nodes of these patients which are also two extra non-identifiable patient labels. For separating the cells based on non-identifiable patient labels (patient ID), Figure 6 reveals that DBSCAN, Agglomerative, and Mean Shift clustering all have distinct different partitioning from the others models in that they create one large cluster and the rest are significantly smaller. This is indicted by the colors, with one color being assigned to many points within the large group of cells. The number of clusters generated by these models is also somewhat different from the rest of the models. These three models have created what seems to be approximately four clusters each, vastly different than 13. The other models all have roughly the same partitioning and size in clusters. Ward, Spectral Clustering, BIRCH, Gaussian Mixture, and K-Means separate the entire population of cells into many subgroups, indicated by the large variety in colors. The groups are all fairly small, and the large group of cells in the middle of the plot is divided up into subgroups with the smaller groups surrounding that large group also differentiated into a different cluster. Affinity Propagation divided the entire population into about five subgroups, splitting the large group of cells into four subgroups and creating a fifth cluster from the small group of cells outside of the large group. With thirteen different groups, there is a lot more variance with partitioning and creating groups.

Figure 4: Clustering for differences in molecular subtype

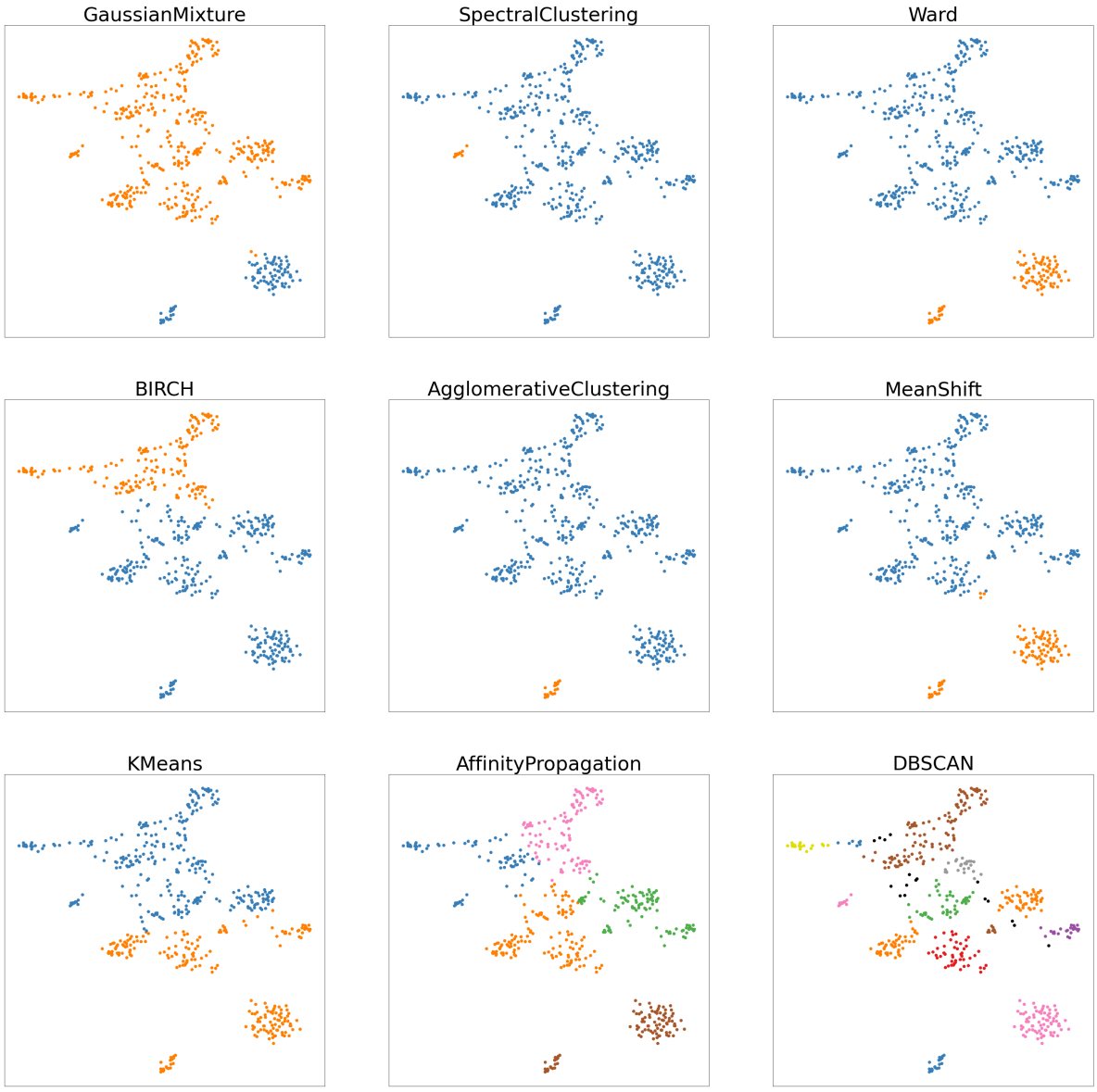


Figure 5: Clustering for differences in cell states, tumor versus non-tumor cells

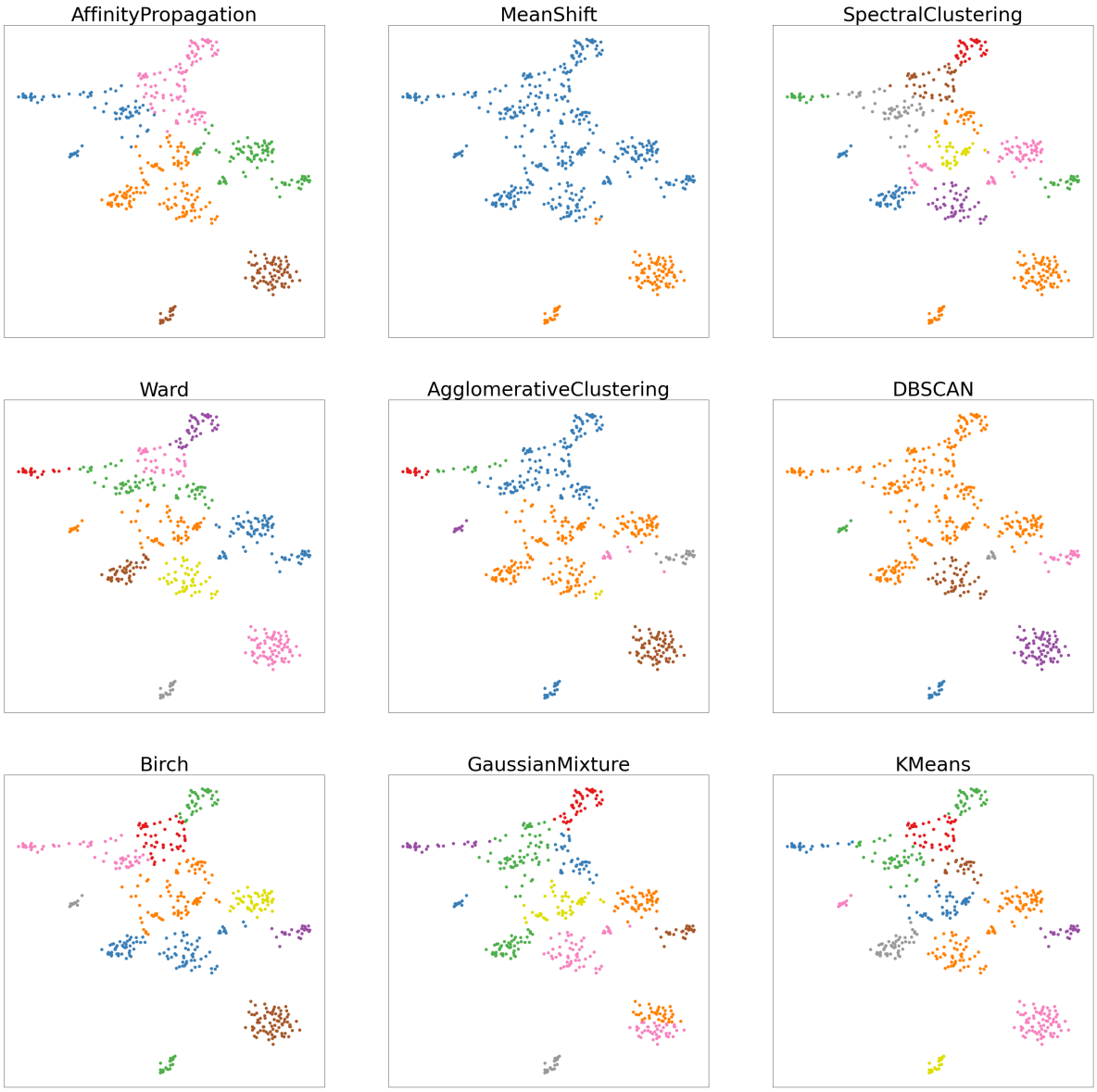


Figure 6: Clustering for on differences in non-identifiable patient labels

2.9 Evaluation Algorithm

2.9.1 Motivation

This evaluation algorithm was created with the intent of assessing how well the clustering algorithm did based on how homogeneous each cluster created was. Overall, it observes the makeup of each cluster, recording the concentration of each cell type in each cluster. A clustering algorithm will have performed well if it has a high concentration of one type of cell in each cluster, and each cluster has a high concentration of a different cell type.

2.9.2 Methods

In the following calculations, the columns of each matrix represent the percentage values of each cluster. The rows of each matrix represent the percentage values of each subtype. Therefore, in the following matrix A, $A_{i,j}$ is the percentage of subtype i in cluster j. For example, in equation 20, with 0 as the first index, $A_{1,1}$ holds the value 1.0 representing 100% of C_2 (cluster 2) is comprised of subtype ERPCELL. In the matrices, the crossed out entries represent the disregarded percentages resulting from their elimination during the evaluation algorithm. The algorithm first searches for the absolute maximum value within the matrix. After identifying that value, each value in the same row and column of that entry is crossed out. The algorithm then searches for the next absolute maximum value among the remaining entries and proceeds to cross out the entries in the same row and column. This process continues until the a number of values reflecting the number of clusters is left.

In this example, DPCELL means double positive cell, ERPCELL means estrogen receptor positive cell, HER2P means human epidermal growth factor 2 cell, and TNBC means triple negative breast cancer. The value k = 4 was chosen because it reflects the actual number of cell populations that were meant to be clustered; there were four different types therefore each clustering method should have created four different clusters representing each type. The example reflects no particular clustering method evaluated in the algorithm but rather shows the general procedure of the evaluation algorithm in both an optimal and suboptimal scenario.

(a) Accuracy Calculation Ideal Case

$$\begin{array}{c} \\ \text{DPCELL} \\ \text{ERPCELL} \\ \text{HER2P} \\ \text{TNBC} \end{array} \begin{array}{c} \begin{array}{cccc} C_1 & C_2 & C_3 & C_4 \end{array} \\ \begin{pmatrix} 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{pmatrix} \end{array} \tag{4}$$

$$\begin{array}{c} \\ \text{DPCELL} \\ \text{ERPCELL} \\ \text{HER2P} \\ \text{TNBC} \end{array} \begin{array}{c} \begin{array}{cccc} C_1 & C_2 & C_3 & C_4 \end{array} \\ \begin{pmatrix} 1.0 & \cancel{0.0} & \cancel{0.0} & \cancel{0.0} \\ \cancel{0.0} & 1.0 & \cancel{0.0} & \cancel{0.0} \\ \cancel{0.0} & \cancel{0.0} & 1.0 & \cancel{0.0} \\ \cancel{0.0} & \cancel{0.0} & \cancel{0.0} & 1.0 \end{pmatrix} \end{array} \tag{5}$$

$$C_1 = \text{DPCELL } (1.0)$$
$$C_2 = \text{ERPCELL } (1.0)$$
$$C_3 = \text{HER2P } (1.0)$$
$$C_4 = \text{TNBC } (1.0) \tag{6}$$

(b) Accuracy Calculation Non-Ideal Case

$$\begin{array}{c} \\ \text{DPCELL} \\ \text{ERPCELL} \\ \text{HER2P} \\ \text{TNBC} \end{array} \begin{array}{c} \begin{array}{cccc} C_1 & C_2 & C_3 & C_4 \end{array} \\ \begin{pmatrix} 0.74 & 0.2 & 0.1 & 0.1 \\ 0.2 & 0.15 & 0.3 & 0.5 \\ 0.03 & 0.2 & 0.6 & 0.4 \\ 0.03 & 0.55 & 0.0 & 0.1 \end{pmatrix} \end{array} \tag{7}$$

$$\begin{array}{c} \\ \text{DPCELL} \\ \text{ERPCELL} \\ \text{HER2P} \\ \text{TNBC} \end{array} \begin{array}{c} \begin{array}{cccc} C_1 & C_2 & C_3 & C_4 \end{array} \\ \begin{pmatrix} 0.74 & \cancel{0.2} & \cancel{0.1} & \cancel{0.1} \\ \cancel{0.2} & \cancel{0.15} & \cancel{0.3} & 0.5 \\ \cancel{0.03} & \cancel{0.2} & 0.6 & \cancel{0.4} \\ \cancel{0.03} & 0.55 & \cancel{0.0} & \cancel{0.1} \end{pmatrix} \end{array} \tag{8}$$

$$C_1 = \text{DPCELL } (0.74)$$
$$C_2 = \text{TNBC } (0.55)$$
$$C_3 = \text{HER2P } (0.6)$$
$$C_4 = \text{ERPCELL } (0.5) \tag{9}$$

The truth dataset in this work refers to the dataset that holds the true cell information for each cell (the clusters are composed of the unlabeled data where the true cell information was unknown). To evaluate the accuracy of the models in this work, the truth dataset is examined and files are

created containing each "true" population or what each cluster should be made of. Then, the cells are compared to every true cell to see what their true label should be. The percentages of what cells (ERPCELL, DPCELL, HER2P, and TNBC), (tumor v. nonTumor), or (BC01, BC02, BC03, BC04, BC05, BC06, BC07, BC08, BC09, BC10, BC11, BC03_LN, BC07_LN) are represented in each cluster will be written to files. Then, the assumption that each cluster is supposed to represent a singular population will be made and each cell population that is the majority of each cluster is marked as correct whereas the remaining cells of that cluster are marked incorrect. The accuracy is represented as the number of "correct" cells over the total number of cells clustered.

Validation for this method of evaluation tests whether the evaluation algorithm correctly identified which clustering methods were high performing. That is, it checked to see that clustering algorithms that had a complete random makeup of cells in each cluster did not receive high "accuracy" scores; it validated that the evaluation algorithm was sensitive to differences in cell type concentration in each cluster. The validation method described in the next paragraph is only a validation method for the evaluation algorithm created, and described previously.

A higher level description of the validation method is depicted as follows. To review, the evaluation method finds the true value of each data point plotted, so it can see what cell types are included in each cluster; it makes the assumption that the highest concentration of cell type presented in the cluster is the cluster's intended group. For example, if 90% of a cluster consists of G2 cells, then that cluster is supposed to be the G2 group. Each cell that does not belong in the cluster takes away from the algorithms "accuracy" score meaning it classifies cells that are of a different type than the highest concentration as "incorrectly" clustered. The validation method randomizes the true labels so that the true value of each data point is incorrect. This makes it so that the evaluation method is looking at clusters with either partially or completely random makeup. It then looks at how the evaluation method "scores" each clustering method even with randomized data. It should give lower accuracy scores to this randomized data.

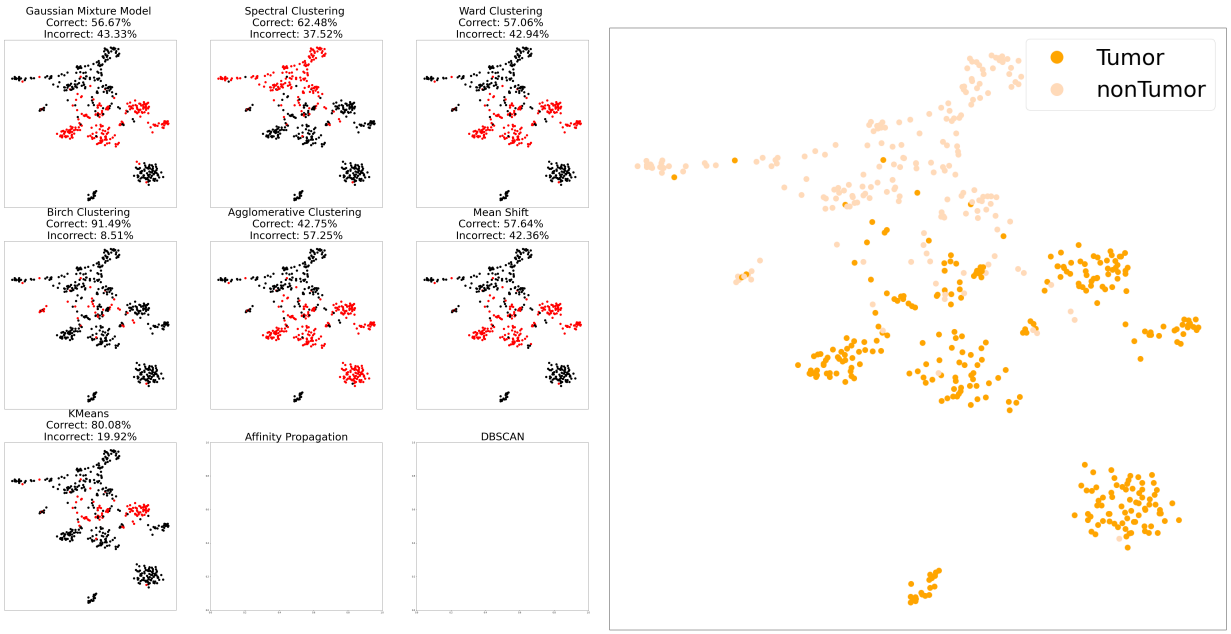
The truth data was completely randomized and partially randomized (10% of the true labels were mixed up); then 100 runs were conducted; the difference in each run was seen only with the evaluation algorithm in the mappings to true cell states. The differences in labels have no effect on clustering. Essentially, the accuracy was initially calculated by extracting the labels of each point plotted, and, separately, finding the true values of each data point (because the true data was accessible in this project). From there, each cluster's contents could be identified by mapping the true data to the clusters. Using the evaluation algorithm, clusters would then be assumed to be a certain type depending on what cells made up the majority of the cluster. By randomizing the truth labels, the mapping of the true values to the clustered cells would be completely modified and result in completely unsuccessful clustering accuracies. If the projected accuracies showed a vast difference with the completely randomized validation set and a slight difference with the partially randomized validation set, then the method for evaluating the algorithms can be deemed efficient. A "vast" difference is considered a substantial change in accuracy such as a difference greater than 30% whereas a "slight" difference is considered a difference between 10% to 20%.

3 Results

We start by showing the visualizations produced through the evaluation algorithm compared to the biological truth of each clustering algorithm. The visualizations are then followed by a brief discussion of the observations. The analysis of 100 runs is then discussed and followed by an explanation of methods used for validation of the evaluation algorithm. In the following diagrams,"Truth v. Predicted" color the cells based on how they reflected the true structure; red points represent

data points that are not supposed to be in the cluster that the model predicted they'd be grouped with and black points represent data points that are supposed to be in that cluster. The empty plots represent models that couldn't cluster the correct number of clusters and were eliminated from further analysis.

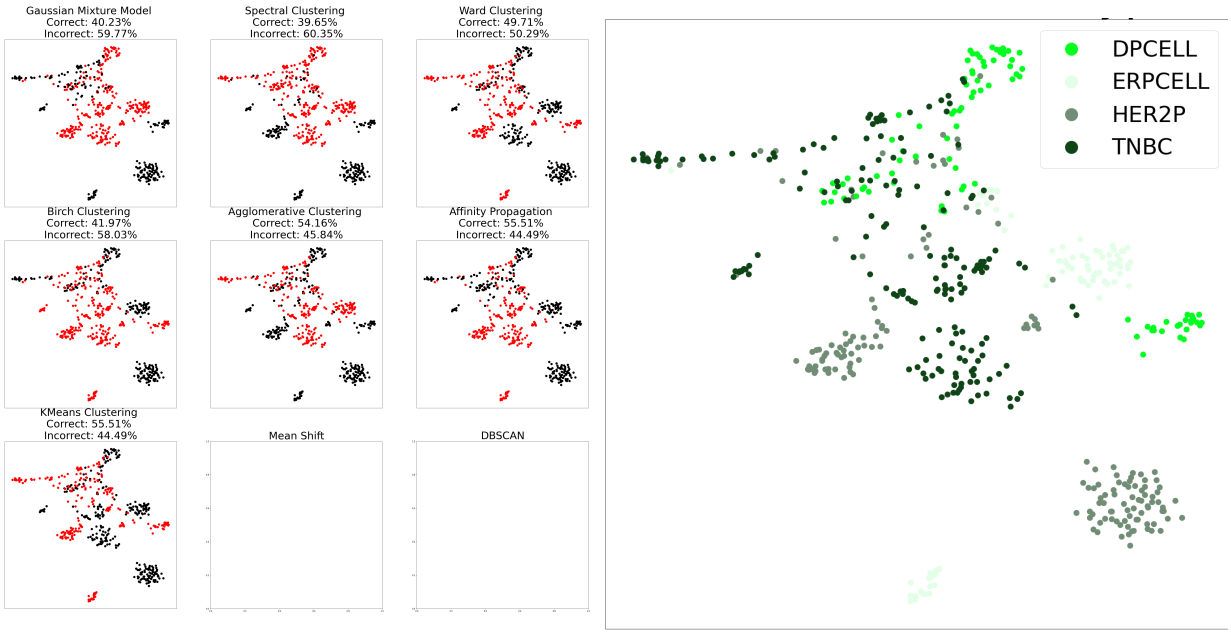
3.1 Visualizations



(a) Ground Truth Compared to Predicted

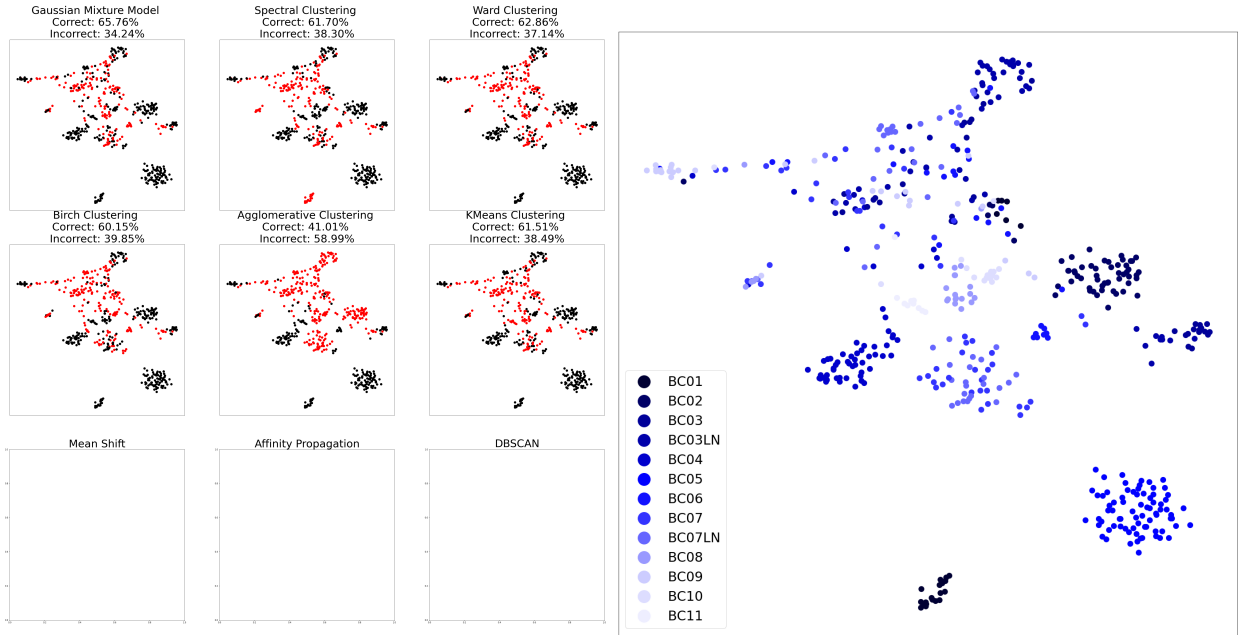
(b) True Tumor Cell Diagram

Figure 8: Tumor Plots



(a) Ground Truth Compared to Predicted (b) True Molecular Subtype Diagram

Figure 9: Molecular Subtype Plots



(a) Ground Truth Compared to Predicted (b) True Patient ID Diagram

Figure 10: Patient ID Plots

For each of the figures, the red points represent the cells that were incorrectly clustered, and the black points represent the cells that were correctly clustered. The evaluation algorithm gauged the homogeneity of each cluster where the cells within the cluster are of the same type. With compari-

son to the truth data, the evaluation algorithm gauges the percentage of each subtype represented in each cluster and iteratively mark which cluster represented which subpopulation based on where the highest concentration of each subtype was. The red data points represent data points that were grouped with dissimilar cells. The black data points represent subgroups within each cluster that should be homogeneous. This means that according to the diversity of cell type within the cluster, these subgroups should be assumed to be of the same type according to what the majority of the labels were in the cluster. Essentially, the black data points represent the cells that were correctly clustered.

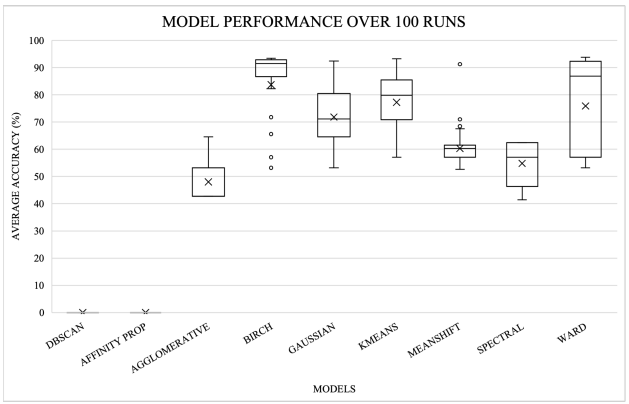
For the tumor plots (Fig. 8), it is visually apparent that for this single run BIRCH and K-Means clustering performed the best with accuracy scores above 80%, with errors primarily on the line of partition between the clusters. The rest of the models were substantially off with their areas of partition which resulted in a lower accuracy score. Most models were able to group together the tumor cells correctly excluding Spectral clustering.

For the molecular subtype plots (Fig. 9), the models performed substantially worse than they did on clustering the tumor cells. The highest performing models for this specific run were K-Means, Agglomerative, and Affinity Propagation clustering. Agglomerative and Affinity Propagation clustering are both very low performing for the other tasks. Regardless, these scores of approximately 55% are vastly lower than the previous scores of above 80% for partitioning the tumor cells. From the visualizations, it is apparent that none of the models did a satisfactory job of accurately clustering the cells.

For the patient ID plots (Fig. 10), all of the models scored between 60% to 65% accuracy excluding Agglomerative Clustering. In this paper, 80% is considered a minimum threshold for a decently successful clustering algorithm; in past studies comparing clustering performance between algorithms [17] [18], this approximate threshold was also used. Considering even the highest performing models scored well below that set threshold, none of them did a strong job of clustering these cells for patient ID.

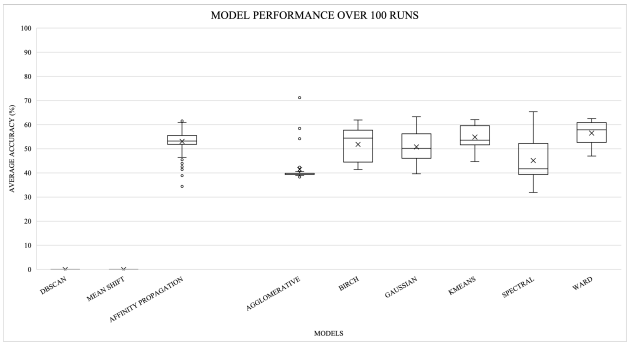
3.2 Analysis

The following plots depict each model's performance–meaning the amount of correct labels clustered over total labels (correct labels can be seen as the black dots in previous figures)–over 100 runs, and 100 runs were conducted due to the stochasticity of each model. For example, some models such as Spectral Clustering, DBSCAN, BIRCH, and others start at an arbitrary point or incorporate randomness within their clustering process. To account for minor changes in performance in these models, the models and evaluation algorithm were run 100 times, and the average accuracy is computed from the 100 produced accuracies.



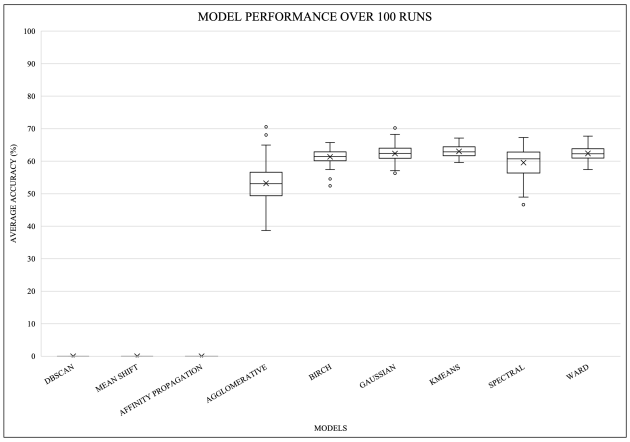
MODEL	AVERAGE ACCURACY (%)	SEM
BIRCH	83.71	1.404
KMEANS	77.26	1.071
WARD	75.89	1.673
GAUSSIAN	71.85	1.063
MEANSHIFT	60.30	0.502
SPECTRAL	54.80	0.080
AGGLOMERATIVE	48.02	0.655
AFFINITY PROP	0.00	0.000
DBSCAN	0.00	0.000

Figure 11: Tumor Cell Plots



MODEL	AVERAGE ACCURACY (%)	SEM
WARD	56.49	0.556
KMEANS	54.85	0.435
AFFINITY PROPAGATION	53.07	0.498
BIRCH	51.82	0.697
GAUSSIAN	50.76	0.575
SPECTRAL	45.11	0.803
AGGLOMERATIVE	41.44	0.579
DBSCAN	0.00	0.000
MEAN SHIFT	0.00	0.000

Figure 12: Molecular Subtype Plots



MODEL	AVERAGE ACCURACY (%)	SEM
KMEANS	63.01	0.168
WARD	62.40	0.231
GAUSSIAN	62.38	0.248
BIRCH	61.34	0.234
SPECTRAL	59.56	0.452
AGGLOMERATIVE	53.20	0.604
AFFINITY PROPAGATION	0.00	0.000
DBSCAN	0.00	0.000
MEAN SHIFT	0.00	0.000

Figure 13: Patient ID Plots

For the tumor cell evaluation (Fig. 11), the highest performing models were BIRCH (83.7%), K-Means (77.3%), and Ward(75.9%) clustering. All of these models performed relatively close to the threshold of 80%. The lowest performing model that generated the correct number of clusters was Agglomerative Clustering (48.0%). DBSCAN and Affinity Propagation failed to separate the total population into the correct number of subpopulations. The models with the most variation were Ward (1.67% Standard Error of the Mean (SEM)) and BIRCH (1.40% SEM), and the models

with the least variation were Spectral Clustering (0.080% SEM) and Mean Shift (0.502% SEM). Based on criteria of high mean accuracy and relatively low variation, K-Means performed well having the second highest accuracy and a median SEM. BIRCH had a vastly higher accuracy than any other model which also qualifies it as a high performing model for the task of classifying tumor cell populations.

For the molecular subtype evaluation (Fig. 12), the highest performing models were Ward (56.5%), K-Means (54.8%), and Affinity Propagation (53.1%). The lowest performing models that generated the correct number of clusters were Spectral Clustering (45.1%) and Agglomerative (41.4%) clustering. DBSCAN and Mean Shift failed to generate the correct number of subpopulations. The models with the highest variation over 100 runs were Spectral Clustering (0.795% SEM) and BIRCH (0.693% SEM); the models with the lowest variation over 100 runs were Affinity Propagation (0.479% SEM) and K-Means (0.402%). On the criteria of high mean accuracy and relatively low variation, both K-Means and Affinity Propagation performed best with the highest accuracies and lowest SEM.

For the patient ID evaluation (Fig. 13), the highest performing models were K-Means (63.0%), Ward (62.4%), Gaussian Mixture Model (62.4%), and BIRCH (61.34%). Affinity Propagation, DBSCAN, and Mean Shift failed to generate the correct number of subpopulations. The lowest performing models were Agglomerative (53.2%) and Spectral Clustering (59.6%). The models with the most variation over 100 runs were Agglomerative (0.598% SEM) and Spectral Clustering (0.451%); the models with the lowest variation were K-Means (0.162% SEM), Ward (0.205% SEM), and BIRCH (0.233% SEM). On the criteria of high mean accuracy and low variation, K-Means, Ward, and BIRCH clustering are considered the best performing models.

3.3 Validation

MODEL	AVERAGE ACCURACY (%)	SEM
BIRCH	82.32	1.130
KMEANS	78.34	0.947
WARD	75.73	1.501
GAUSSIAN	71.56	1.017
MEANSHIFT	59.90	0.465
SPECTRAL	54.21	0.847
AGGLOMERATIVE	49.08	0.955
AFFINITY PROPAGATION	0.00	0.000
DBSCAN	0.00	0.000

(a) Partial Randomization Evaluation

MODEL	AVERAGE ACCURACY (%)	SEM
SPECTRAL	57.68	0.697
AGGLOMERATIVE	53.05	0.928
KMEANS	49.64	0.316
MEAN SHIFT	48.22	0.729
BIRCH	48.07	0.401
GAUSSIAN MIXTURE	46.53	0.282
WARD	46.10	0.346
AFFINITY PROPAGATION	0.00	0.000
DBSCAN	0.00	0.000

(b) Complete Randomization Evaluation

Figure 14: Tumor Cell Clustering Evaluation Tables

MODEL	AVERAGE ACCURACY (%)	SEM
WARD	53.14	0.743
GAUSSIAN	51.41	0.667
KMEANS	50.97	0.463
AFFINITY PROPAGATION	48.81	0.546
BIRCH	48.10	0.729
SPECTRAL	43.96	0.769
AGGLOMERATIVE	41.40	0.562
DBSCAN	0.00	0.000
MEAN SHIFT	0.00	0.000

(a) Partial Randomization Evaluation

MODEL	AVERAGE ACCURACY (%)	SEM
AGGLOMERATIVE	32.23	0.469
GAUSSIAN	29.41	0.188
WARD	28.99	0.112
SPECTRAL	28.72	0.380
KMEANS	28.69	0.061
AFFINITY PROPAGATION	28.48	0.294
BIRCH	28.00	0.221
DBSCAN	0.00	0.000
MEAN SHIFT	0.00	0.000

(b) Complete Randomization Evaluation

Figure 15: Molecular Subtype Cell Clustering Evaluation Tables

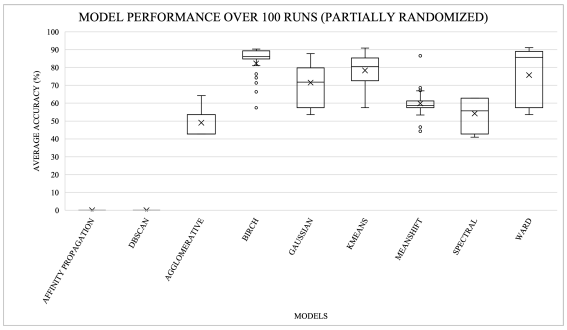
MODEL	AVERAGE ACCURACY (%)	SEM
KMEANS	57.15	0.193
WARD	56.12	0.454
BIRCH	55.48	0.400
GAUSSIAN	55.11	0.363
SPECTRAL	53.03	0.475
AGGLOMERATIVE	48.09	0.637
AFFINITY PROPAGATION	0.00	0.000
DBSCAN	0.00	0.000
MEAN SHIFT	0.00	0.000

(a) Partial Randomization Evaluation

MODEL	AVERAGE ACCURACY (%)	SEM
GAUSSIAN	14.03	0.165
SPECTRAL	13.79	0.162
KMEANS	13.60	0.109
WARD	13.33	0.102
BIRCH	12.85	0.151
AGGLOMERATIVE	10.22	0.199
AFFINITY PROPAGATION	0.00	0.000
DBSCAN	0.00	0.000
MEAN SHIFT	0.00	0.000

(b) Complete Randomization Evaluation

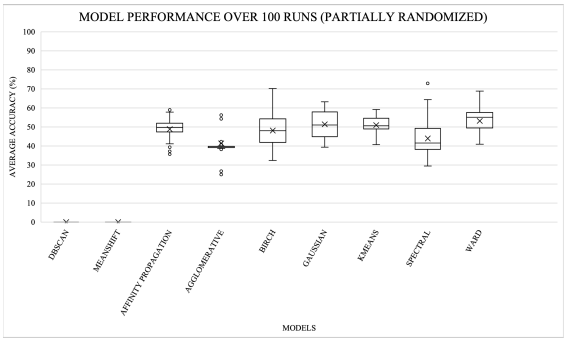
Figure 16: Patient ID Cell Clustering Evaluation Tables



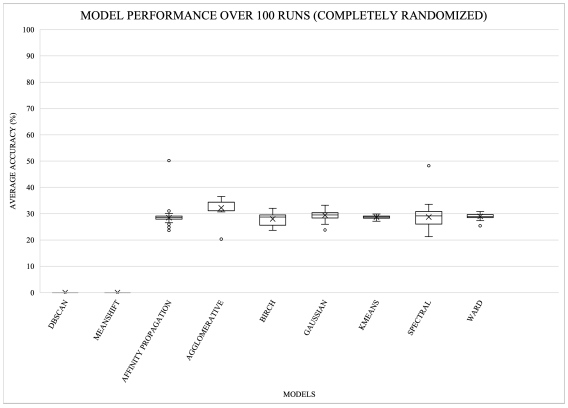
(a) Partial Randomization Evaluation

(b) Complete Randomization Evaluation

Figure 17: Tumor Cell Clustering Evaluation Plots

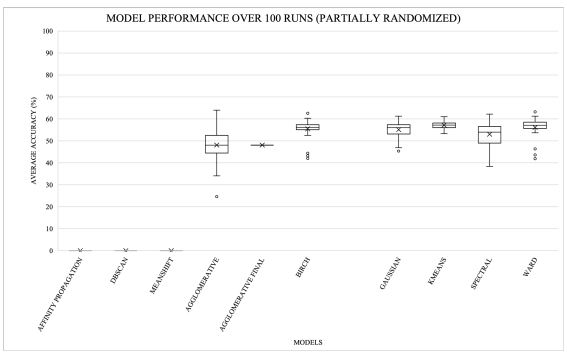


(a) Partial Randomization Evaluation

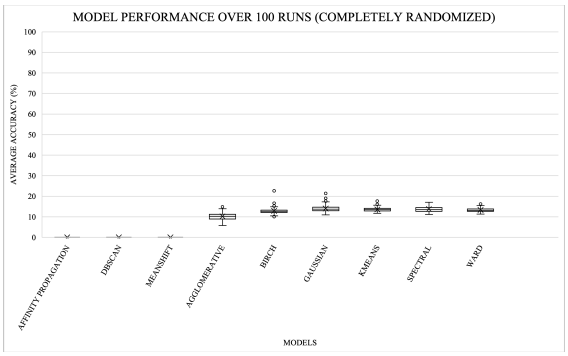


(b) Complete Randomization Evaluation

Figure 18: Molecular Subtype Cell Clustering Evaluation Plots



(a) Partial Randomization Evaluation



(b) Complete Randomization Evaluation

Figure 19: Patient ID Cell Clustering Evaluation Plots

Ideally, a robust evaluation algorithm would be sensitive to changes in the truth datasets. That is, the evaluation algorithm should give high accuracy scores to clusters that have high concentrations of certain cell types as opposed to clusters with a random cellular makeup. With partial randomization, the accuracy scores for each model should drop slightly compared to no randomization at all. With complete randomization, the accuracy scores should drop largely compared to no randomization.

The randomization tests shown above serve as a method to validate the method of evaluation for the algorithms. Randomization refers to randomizing the truth labels themselves by mapping each data point to a different label than the true label. The evaluation algorithm is expected to produce lower scores than initially calculated because the truth data is completely different than the model predictions. However, it is likely each accuracy score will still be above 0% for complete randomization because there is a chance that after randomization some cells still have their true labels; there would just be a substantially fewer number of them. There were two levels of randomization: partial and complete randomization. Partial randomization means that 10% of the true labels were randomly selected and then re-mapped to different cells. Complete randomization refers to when all true labels were re-mapped to different cells. After randomization, 100 runs of the clustering and evaluation algorithm were run again with the average accuracy computed from the 100 accuracies produced.

For each task, complete randomization did result in substantially lower scores. Generally, a model scored approximately $\frac{1}{n}\%$ (where n is the number of subgroups) for complete randomization. Partial randomization showed a slightly lower range of scores for each model with a drop in scores of about 5%-10% for each model.

4 Discussion

We investigated the efficacy of unsupervised analysis on scRNAseq data through a pipeline and evaluation metrics to rank nine models (BIRCH, Ward, Spectral Clustering, Gaussian Mixture Model, DBSCAN, Affinity Propagation, Agglomerative Average Linkage, Mean Shift, and K-Means). With one dataset, we sought to evaluate these models on their ability to separate subclonal cell populations represented within the data. These subpopulations were cell state (tumor versus nonTumor), molecular subtype (estrogen receptor positive, estrogen receptor positive, double positive, triple negative breast cancer, and human epidermal growth factor), and patient identification (BC01, BC02, BC03, BC03LN, BC04, BC05, BC06, BC07, BC07LN, BC08, BC09, BC10, BC11). Unsupervised learning was used as opposed to supervised learning because, although

truth data was accessible in this research, typically scRNAseq data is unlabeled. This research is meant to evaluate techniques commonly used in scRNAseq data analysis practice.

In this research, we found that unsupervised learning is effective in single cell RNA sequencing data projection and analysis. For the tumor cell identification tasks, some models were fairly successful, clustering approximately 80% of the data points correctly. However, for the other tasks (patient ID and molecular subtype), the models did not show high performance, reaching a maximum of approximately 60% accuracy. Although accuracy varied depending on task, most models would effectively distinguish the correct number of subpopulations with the exception of DBSCAN, Affinity Propagation, and Mean Shift. Of these models, K-Means, BIRCH, and Ward clustering have the highest mean accuracy and low variation over 100 runs for each task. BIRCH excelled at classification of tumor cell data with an average accuracy above 80% while K-Means and Ward performed best on both patient ID and molecular subtype classification. Dimensionality reduction with PCA and then preliminary visualization with UMAP are effective techniques for reducing the number of components in the dataset to effectively cluster the data points. By previewing the biological mappings, it can be seen that biological structure preserved in that distinct partitions between subtypes were created.

Affinity Propagation, DBSCAN, and Mean Shift are all models that do not take the number of desired clusters as an input to the model; because of this, they were more likely to be unsuccessful in distinguishing the correct number of subpopulations as opposed to other models that had n_clusters as a parameter. BIRCH and Ward are both forms of agglomerative hierarchical clustering that start with a single or small number of observations and build clusters by continually merging similar observations together. K-Means, although not a form of centroid based clustering, also starts at a random observation and attempts to find similar data points until it can find a point that is best suited to be a centroid. These methods may have performed best because they examined each cell on the individual level first and worked to build a hierarchy or certain groups iteratively as opposed to other methods such as Agglomerative Average Linkage and Spectral Clustering which look at data points on a global scale initially.

BIRCH scoring above 80% on average for accuracy in clustering tumor cell data shows that it is effective in clustering tumor v. nontumor cells. However, for the rest of the tasks (patient ID and molecular subtype), the models often performed at an accuracy of about 60% which is below the typical performance target. Of these models, K-Means and Ward were the highest performing. However, this does not mean that these should be used as is for the tasks specifically. Often, the method that Ward and K-Means have for partitioning the data would respond best to the structure, yet there is still a large room for improvement. This can also be an indication that gene expression levels are not the best way of classifying molecular subtype or patient ID, for the models were not able to recognize distinct differences in the levels to strongly cluster the data. An optimized pipeline using BIRCH, K-Means, and Ward clustering for each task along with proper parameterization for UMAP and PCA would prove to be as effective as possible for visualizing cell subpopulations. Although not entirely accurate, BIRCH, K-Means, and Ward clustering were the highest performing models from the runs of the pipeline that show potential to generate proper clusters of tumor v. nonTumor cells, patient ID, and molecular subtype which may help recognize different phenotypes within cell populations.

Although the clustering models were not completely successful in the analysis tasks, some models such as BIRCH, K-Means, and Ward had a fairly high accuracy on the tumor cell classification task. Other tasks (patient ID and molecular subtype) requiring more subgroup identification were a challenge for all of the models. While computational tools are still improving, its important to recognize tasks in which certain models show high performance. In this paper, tumor cell identification is a task where models performed successfully whereas patient ID and molecular subtype

identification still requires more investigation on which models are best suited.

An evaluation of the most frequently used models is particularly important, for there has not been an in depth discussion of how these unsupervised methods compare to each other for scRNAseq data analysis. In a clinical setting, this optimized pipeline can be used to improve scRNAseq analysis because it incorporates the most efficient and effective techniques found in this study. The pipeline proposed in this study is: data wrangling; dimensionality reduction with PCA; preliminary visualization with UMAP; and clustering with BIRCH, K-Means, and Ward clustering. By using this pipeline, clinicians can learn more about the nature of malignant cell growth and tumor heterogeneity which directly affects the quality of cancer treatment. They would be able to visualize the output of the scRNAseq dataset, creating a foundation for later visualization of subgroups in cell populations with these models having the best chance of distinguishing these groups correctly.

5 Acknowledgements

This work is supported by the NIH NCI specifically from the grant NIH-CSBC: U54 CA209978. I would like to thank my mentor Dr. Frederick R. Adler at the University of Utah Math Department, and Dr. Jim Waisman and team for the supply of clinical data. I would also like to thank Dr. Jeffrey Chang and his group for assembling the data and Dr. Mark Smithson for help during the initial phases of this work.

References

[1] URL: http://proceedings.mlr.press/v119/fan20b/fan20b.pdf.

[2] 14.7 - ward's method: Stat 505. https://online.stat.psu.edu/stat505/lesson/14/14.7.

[3] 2.1. gaussian mixture models. URL: https://scikit-learn.org/stable/modules/mixture.html.

[4] Interpret the key results for principal components analysis. https://support.minitab.com/en-us/minitab/18/help-and-how-to/modeling-statistics/multivariate/how-to/principal-components/interpret-the-results/key-results/.

[5] Hierarchical clustering, September 2008. URL: https://www.stat.cmu.edu/~cshalizi/350/2008/lectures/08/lecture-08.pdf.

[6] Hierarchical clustering, Feb 2016. URL: https://www.solver.com/xlminer/help/hierarchical-clustering-intro.

[7] Dbscan with math. https://medium.com/@rdhawan201455/dbscan-clustering-algorithm-with-maths-c40dcee88281, November 2019.

[8] Mean-shift clustering. https://www.geeksforgeeks.org/ml-mean-shift-clustering/, May 2019.

[9] Adam Pearce Andy Coenen. A dip into umap theory. https://pair-code.github.io/understanding-umap/.

[10] Alakesh Bora. Ml birch clustering. https://www.geeksforgeeks.org/ml-birch-clustering/, July 2020.

[11] M. Brems. A one-stop shop for principal component analysis. https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c, 2019.

[12] Neerja Doshi. Spectral clustering, May 2019. URL: https://towardsdatascience.com/spectral-clustering-82d3cff3d3b7.

[13] Ahmad Fouad El-Samak and Wesam M. Ashour. Optimization of traveling salesman problem using affinity propagation clustering and genetic algorithm, Jan 1970. URL: https://iugspace.iugaza.edu.ps/handle/20.500.12358/24581?locale-attribute=en.

[14] William Fleshman. Spectral clustering, Feb 2019. URL: https://towardsdatascience.com/spectral-clustering-aba2640c0d5b.

[15] Jared Gerschler. The dbscan algorithm background and theory. https://jaredgerschler.blog/2017/01/13/the-dbscan-algorithm-background-and-theory/, 2017.

[16] B. Hwang, J. H. Lee, and D. Bang. Single-cell RNA sequencing technologies and bioinformatics pipelines. *Exp Mol Med*, 50(8):1–14, 08 2018.

[17] Yong Gyu Jung, Min Soo Kang, and Jun Heo. Clustering performance comparison using k-means and expectation maximization algorithms. *Biotechnology, biotechnological equipment*, 28(sup1):S44–S48, Nov 2014. 26019610[pmid]. URL: https://pubmed.ncbi.nlm.nih.gov/26019610, https://doi.org/10.1080/13102818.2014.949045 doi:10.1080/13102818.2014.949045.

[18] Mudassir Khan. Kmeans clustering for classification, Aug 2017. URL: https://towardsdatascience.com/kmeans-clustering-for-classification-74b992405d0a.

[19] S. Kolchenko. Machine learning meets biology (once again) or single cell rna sequencing as a field for unsupervised learning. https://sergeykolchenko.medium.com/machine-learning-meets-biology-once-again-or-single-cell-rna-sequencing-as-a-field-for-2be24f9108dd, 2019.

[20] Jasmine A. McQuerry, Jeffrey T. Chang, David D. L. Bowtell, Adam Cohen, and Andrea H. Bild. Mechanisms and clinical implications of tumor heterogeneity and convergence on recurrent phenotypes. *Journal of Molecular Medicine*, 95(11):1167–1178, Nov 2017. https://doi.org/10.1007/s00109-017-1587-4 doi:10.1007/s00109-017-1587-4.

[21] Matt Nedrich. Mean shift clustering. https://spin.atomicobject.com/2015/05/26/mean-shift-clustering/, 2015.

[22] Nikita Sharma. Understanding the mathematics behind k-means clustering. https://heartbeat.fritz.ai/understanding-the-mathematics-behind-k-means-clustering-40e1d55e2f4c?gi=d3462e3dc02f, February 2020.

[23] Aishwarya Singh. Build better and accurate clusters with gaussian mixture models. https://www.analyticsvidhya.com/blog/2019/10/gaussian-mixture-models-clustering/, 2019.

[24] Giorgio Stanta and Serena Bonin. Overview on clinical relevance of intra-tumor heterogeneity. *Frontiers in Medicine*, 5:85, 2018. URL: https://www.frontiersin.org/article/10.3389/fmed.2018.00085, https://doi.org/10.3389/fmed.2018.00085 doi:10.3389/fmed.2018.00085.

[25] Harshita Vemula. How affinity propagation works. https://towardsdatascience.com/math-and-intuition-behind-affinity-propagation-4ec5feae5b23, August 2019.

[26] L. Yan, N. Rosen, and C. Arteaga. Targeted cancer therapies. *Chin J Cancer*, 30(1):1–4, Jan 2011.

[27] Soner Yıldırım. Dbscan clustering-explained, Apr 2020. URL: https://towardsdatascience.com/dbscan-clustering-explained-97556a2ad556.

[28] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: An efficient data clustering method for very large databases. 1996.