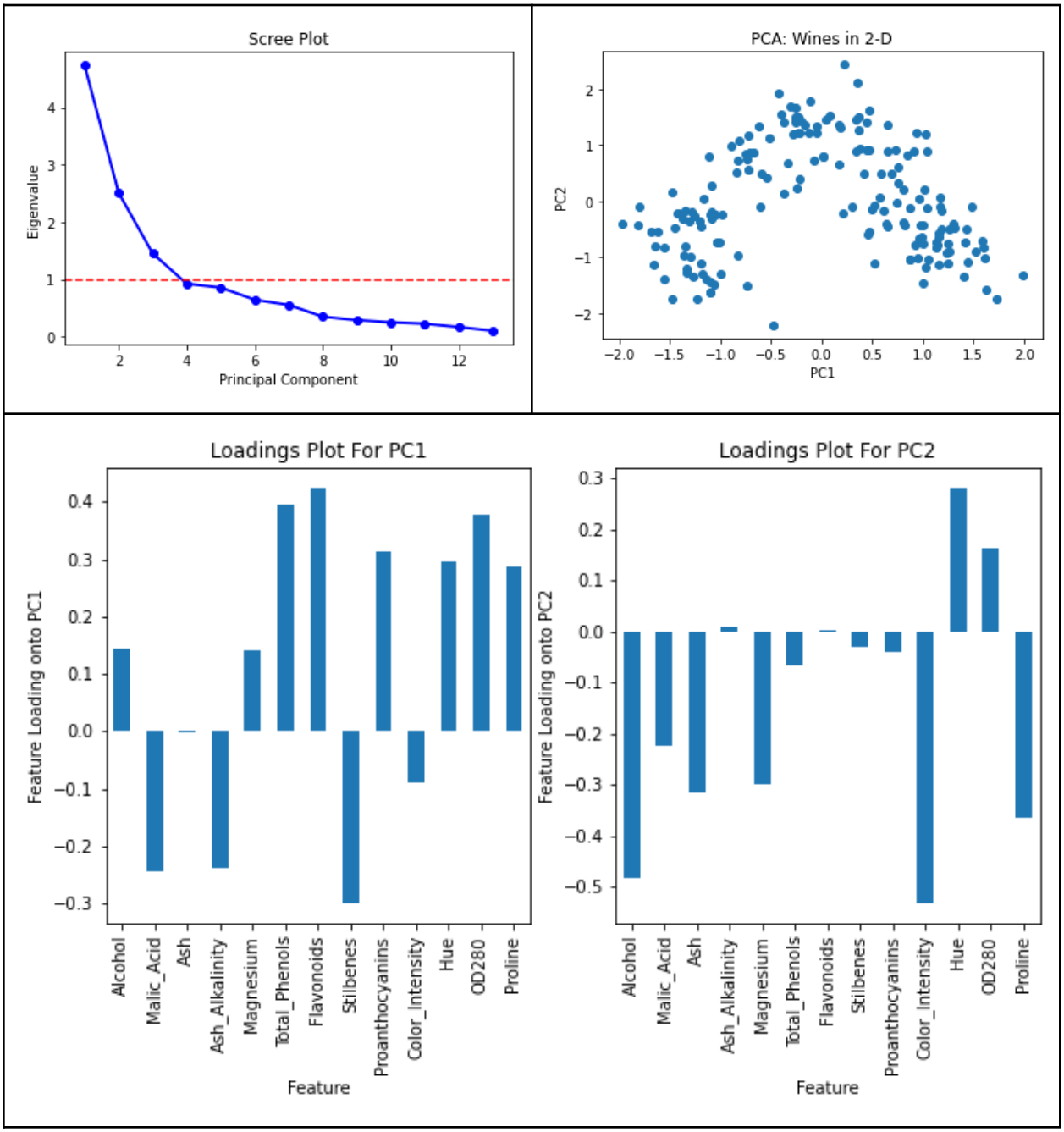
Simone Rittenhouse
Prof. Pascal Wallisch
Into to Machine Learning
11 May 2022

Intro to Machine Learning: Homework 5

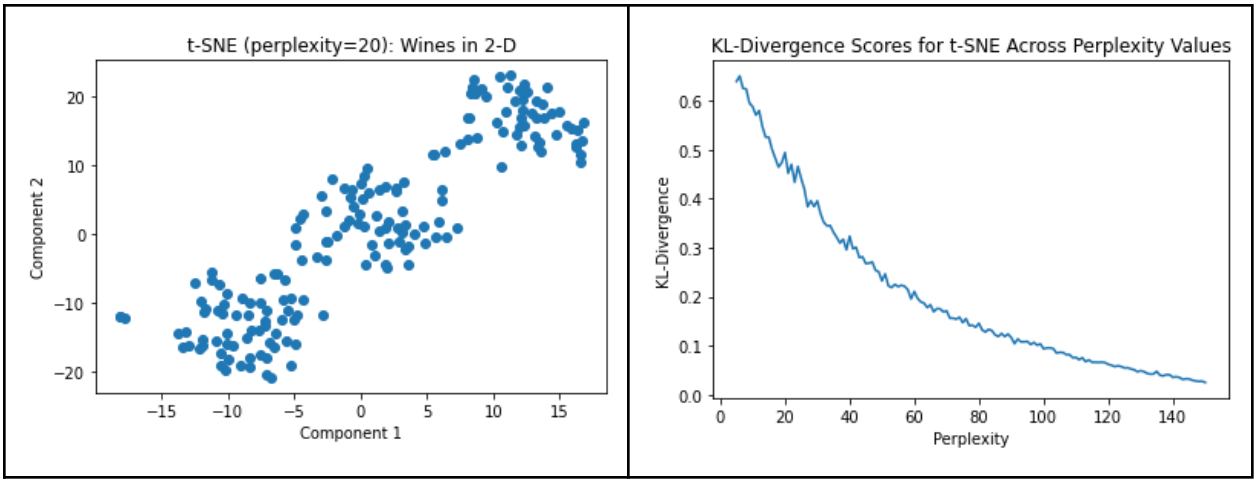
1. **Do a PCA on the data. How many Eigenvalues are above 1? Plotting the 2D solution (projecting the data on the first 2 principal components), how much of the variance is explained by these two dimensions, and how would you interpret them?**
    a. Before running a PCA on the data, I normalized the original dataset by z-scoring. I then ran a PCA using sklearn.decomposition's PCA object. I fit this object on the normalized dataset and then created a new dataset by transforming the original through the trained PCA. I used the explained_variance_ attribute to find the eigenvalues, and I plotted a Scree plot to examine which eigenvalues were greater than one. I then plotted a 2D solution by taking the first two columns of the transformed dataset (corresponding to the first two principal components) and creating a scatter plot. Lastly, I found the amount of variance explained by these two components using the PCA object's explained_variance_ratio and created loadings plots for both using the PCA components_ attribute to interpret the two components.
    b. I first normalized the original dataset because, without demeaning the data, the first principal component would have pointed towards the mean. Without standardizing the data, the PCA would have been dominated by features with high variance - making features with high variability count more than others. I used explained_variance_ to get the eigenvalues because the documentation states that this attribute is the largest eigenvalue for n_components (since n_components was set to be all features this would return all eigenvalues). A Scree plot was used to show which eigenvalues are above one since this plot looks at the magnitude of the eigenvalues for all components, and a 2D solution was plotted using the first two principal components since these explain the most variance in the dataset. To find the amount of variance explained by the first two principal components, the explained_variance_ratio was used because the documentation states that the attribute returns the amount of variance explained by the selected components. Lastly, plotting the loadings for the two first principal components was used to help me interpret what information they represent as the loadings are the weight of each feature in the principal component and show how much each variable contributed (either positively or negatively).
    c. Based on the Scree plot shown below, three eigenvalues were above one. The scatter plot showing the data in 2-dimensions also seems to suggest that there may be three clusters of data in 2D space. For the first two principal components, I

found that they explained 36.2% and 19.21% of the variance in the data respectively, for a total of 55.41% of the variance. Lastly, the loadings plot for the first principal component shows that total_phenols, flavonoids, and OD280 had the largest positive weight while stilbenes had a larger negative weight - suggesting this feature is negatively correlated with the component. For the second principal component, hue had the largest positive weight; however, most of the variables had large negative correlations with PC2. The largest overall associations were with alcohol, color intensity, and proline - which were negative.



d. I found that three eigenvalues out of 13 were over 1, and the first two principal components explained over half of the total variance of the data (55.41%). This makes sense because each component tries to maximize the variance along a new orthonormal basis, so the first principal components will explain the most variance in the dataset. Additionally, the scatter plot shows three possible clusters when the dataset is projected onto the first two principal components. Lastly, the loadings plots of these components suggest that the first component may be interpreted as primarily capturing the total phenols, flavonoids, and OD280 in wines. The second component had the largest relationships with alcohol, color intensity, and proline, although these features have a negative correlation with the component - meaning that traveling positively up the y-axis of our scatter plot corresponds to lower values of these features in the original dataset.

2. **Use t-SNE on the data. How does KL-divergence depend on Perplexity (vary Perplexity from 5 to 150)? Make sure to plot this relationship. Also, show a plot of the 2D component with a Perplexity of 20.**
    a. To use t-SNE on the data, I again used the normalized dataset from question one. I then iterated over perplexity values from 5 to 150, using sklearn.manifold's TSNE object to apply t-SNE to the data (using the fit_transform method). I then found the KL-divergence of each perplexity value by using the kl_divergence_ attribute. I plotted the KL-divergence values across all values of perplexity tested, as well as plotting a scatter plot of the first two components of the transformed dataset for a perplexity value of 20.
    b. I used the normalized dataset as t-SNE may also be disproportionately affected depending on the variance of features (with highly variable features having a greater impact). I then iterated over all values of perplexity in the stated range to see how KL-divergence would depend on this value. I plotted the KL-divergence across perplexity values to visualize their relationship and examine any trends between them. The kl_divergence_ attribute was used since the documentation states this returns the KL-divergence after optimization (i.e. after the t-SNE is fit to the dataset). I also plotted the first two columns of the transformed dataset using a perplexity value of 20 to visualize the data in 2D as these are the first two components of the t-SNE transformed data.
    c. As the scatter plot of the first two components after running t-SNE with perplexity = 20 shows, there are three distinct clusters of data. Additionally, the plot of KL-divergence values shows decreasing KL-divergence as perplexity increases. In fact, there was a KL-divergence of almost 0 at a perplexity of 150, while a perplexity of 5 had a KL-divergence above 0.6. The KL-divergence of the t-SNE using perplexity = 20 was 0.4942.



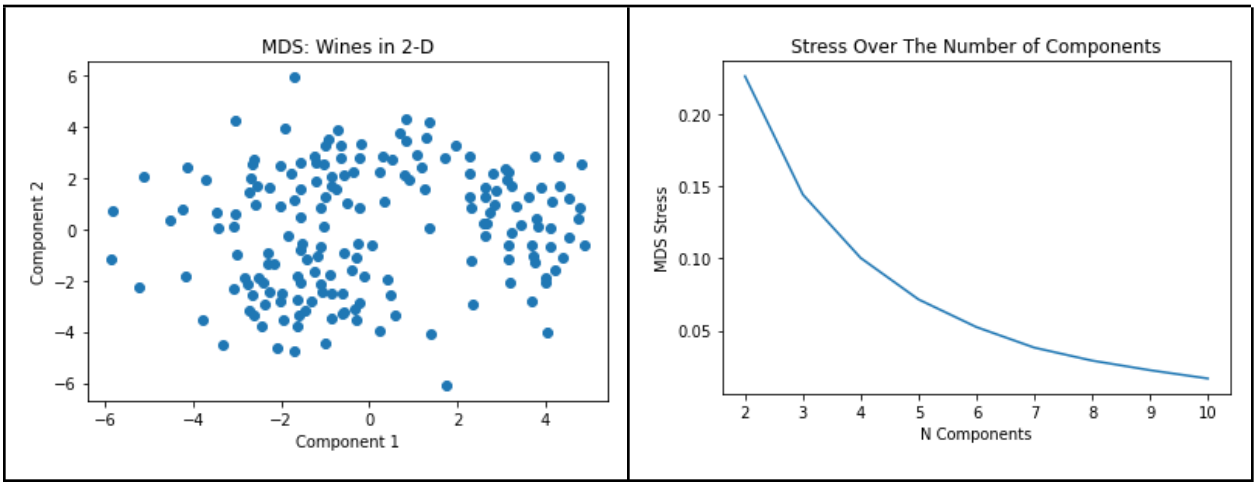
d. My results show that as perplexity increases, KL-divergence decreases. This is likely because perplexity measures the number of points to consider when computing similarity. For a perplexity value of 150, we are essentially considering the entire dataset, meaning that the probability distribution of the data is left relatively unchanged. This would result in a lower KL-divergence value, since the original and transformed probability distributions of the data would be similar. After plotting the 2D transformed dataset using a perplexity of 20, we observe three distinct clusters of data. Unlike PCA, we are unable to easily interpret the components of t-SNE, so it remains unclear what factors contribute to the separation of the dataset into these three categories. Lastly, the KL-divergence value of 0.4942 for the transformed data using perplexity = 20 means that the probability distributions of the original and transformed dataset differ. However, because KL-divergence is unbounded, a value of 0.4942 shows a relatively small divergence between the two probability distributions.

3. **Use MDS on the data. Try a 2-dimensional embedding. What is the resulting stress of this embedding? Also, plot this solution and comment on how it compares to t-SNE**
   a. To use MDS, I first used sklearn.metrics.pairwise's euclidean_distance to transform the normalized dataset into a distance matrix. I then used sklearn.manifold's MDS object to transform the distance matrix. Specifically, I set n_components to 2 for a 2-dimensional embedding, n_init to 100, max_iter to 10,000, and dissimilarity to 'precomputed' since I had already computed the pairwise distance between features. I then used a scatter plot to visualize the data in two dimensions and compare this solution to t-SNE. I found the stress by using the stress attribute of the MDS object after fitting the distance matrix. However, because this attribute does not measure Kruskal's stress but instead an

unnormalized version (from the documentation, the 'sum of squared distance of the disparities and the distances for all constrained points'), Kruskal's stress was computed using the formula 'np.sqrt(mds.stress_ / (0.5 * np.sum(dist_euclid**2)))' where dist_euclid is the distance matrix that was used to optimize the MDS. I repeated this process while increasing the number of embedding components from 2 to 10 to assess how stress varies as a function of the number of dimensions embedded and plotted this relationship to visualize.

b. I first transformed the data using euclidean distance so that the matrix inputted into MDS measured similarity/dissimilarity. Because of this, dissimilarity in the MDS was set to 'precomputed.' The number of components was set to 2 to embed the data in a 2-dimensional space. I set n_init to 100 so that the algorithm was randomly initialized at 100 different positions in low-dimensional space, and max_iter was set to 10,000 to iteratively move the points in lower dimensional space to reduce stress many times at each initialization. I used the formula stated above to compute Kruskal's stress as this is the computation discussed in lecture. Lastly, I tested different numbers of components because I was interested in understanding how stress would change as the data is embedded in higher-dimensional spaces.

c. The two dimensional embedding of the normalized data is less distinctly clustered than the PCA or t-SNE solutions; however, three clusters are still plausible for this data. The resulting stress of the embedding was 0.22613. Additionally, the stress decreased as I increased the number of components.



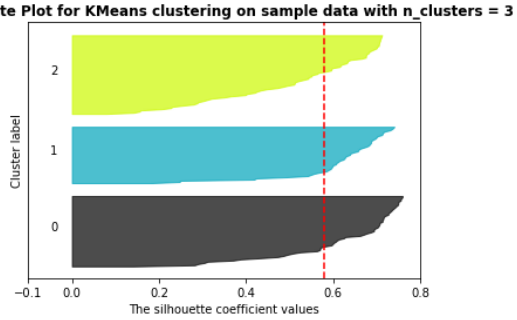
d. I found that the Kruskal stress of a 2-dimensional embedding was 0.22613. This indicates a reasonably good fit for the data, although the stress also decreased as the number of components increased. This is likely because as we embed in higher-dimensional spaces there will be less distance between the original high-dimensional data and the lower-dimensional space MDS embeds into, which

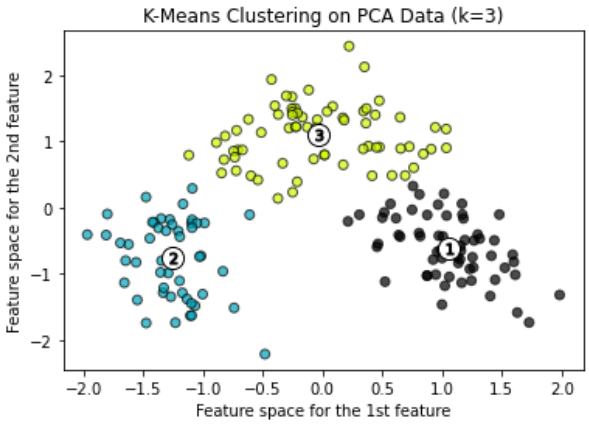
would bring stress to 0. The plot of the 2-dimensional data shows that there are much less distinct clusters than observed with t-SNE or PCA. When compared with t-SNE in particular, we can see that the data is much tighter using MDS and scaled between -6 and 6 instead of -15 and 15. However, MDS also produced less separable clusters, while t-SNE produced the most distinct groups.

4. **Building on one of the dimensionality reduction methods above that yielded a 2D solution (1-3, your choice), use the Silhouette method to determine the optimal number of clusters and then use kMeans with that number (k) to produce a plot that represents each wine as a dot in a 2D space in the color of its cluster. What is the total sum of the distance of all points to their respective clusters centers of this solution?**
    a. For this question, I chose to use the PCA-transformed data. I tested a number of clusters ranging between 2 and 10. For each, the data was given cluster-labels using the fit_predict method of sklearn's KMeans object. The average silhouette score of each clustering was calculated using sklearn.metric's silhouette_score between the original PCA data and the predicted cluster labels. In addition to finding the average silhouette score, I produced silhouette plots for each number of clusters by finding the silhouette score for each point in the PCA dataset and then aggregating and sorting the scores, which were then plotted and labeled with their respective cluster number. Lastly, I examined my output from the average total silhouette scores and silhouette plots, and decided that the optimal number of clusters was three. I used sklearn's KMeans with n_clusters = 3, n_init = 100, and max_iter = 10,000 on the PCA data to predict cluster labels. The data was visualized using a scatter plot where each point was colored based on its cluster and cluster centers were labeled with the cluster's number. The total sum of the distance of all points to their respective cluster centers was found first using the inertia attribute of the KMeans object (which measures total squared distance), and then manually by iterating through the clusters and finding the euclidean distance between each point and the cluster center.
    b. I tested cluster numbers in the range 2 to 10 because I wanted to act under the assumption that I couldn't visually detect how many clusters are optimal for the dataset, though I suspected an optimal number might be three. I found the average silhouette score in order to see in general how ideal each classification is - as a higher silhouette score would indicate a better overall clustering. I also made silhouette plots to see the individual silhouette scores of each point within a cluster. These plots allowed me to see if the majority of the points in a cluster were ideally classified, or if the average was affected by outliers. After inspecting the plots, I chose the optimal cluster number as 3 since it had both the highest average silhouette score and the silhouette plot seemed the most promising. I then

showed a scatter plot of this solution for the purpose of visually inspecting how well the data appeared to be clustered. I used the inertia attribute for the total distance of each point to its cluster center as the documentation states this is the sum of squared distance between points and their cluster centers. To find this value without squaring distance, I manually iterated through clusters and found the euclidean distance between all points in the cluster and its center.

c. I found that the KMeans with n_clusters = 3 had the highest average silhouette score of 0.57953. Additionally, the silhouette plot of this clustering shows that the three clusters have many points with silhouette scores above the average - implying that most points are ideally-classified. The three clusters were also of similar size with few points having low silhouette score values and no points misclassified (i.e. with a silhouette score < 0). The scatter plot of this solution shows three distinct clusters of similar size, which is what I had predicted visually inspecting the unclustered data in question 1. Additionally, the inertia of the clustering was 75.66, and my manual computation yielded a total distance of 104.806 - although I'm suspicious of this value as it should not be higher than the total squared distance.





To see all nine silhouette plots produced, please see the **Appendix** at the end of this document.

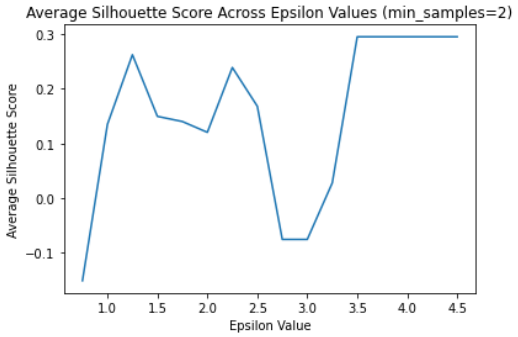
d. Using the silhouette method, I found that the optimal number of clusters for the PCA-transformed data was three. This amount of clusters had both the highest average silhouette score (0.57953) and the most promising silhouette plot, as determined by the large number of individual points in each cluster with high silhouette scores and the similar size of each cluster. The average silhouette score being close to 1 indicates that most points are ideally classified since a value of 0 means arbitrary classification and 1 is perfect classification. The scatter plot of this solution affirms that the points are well classified, as the labeled clusters appear to be plausible. Additionally, the total squared distance of each point to its

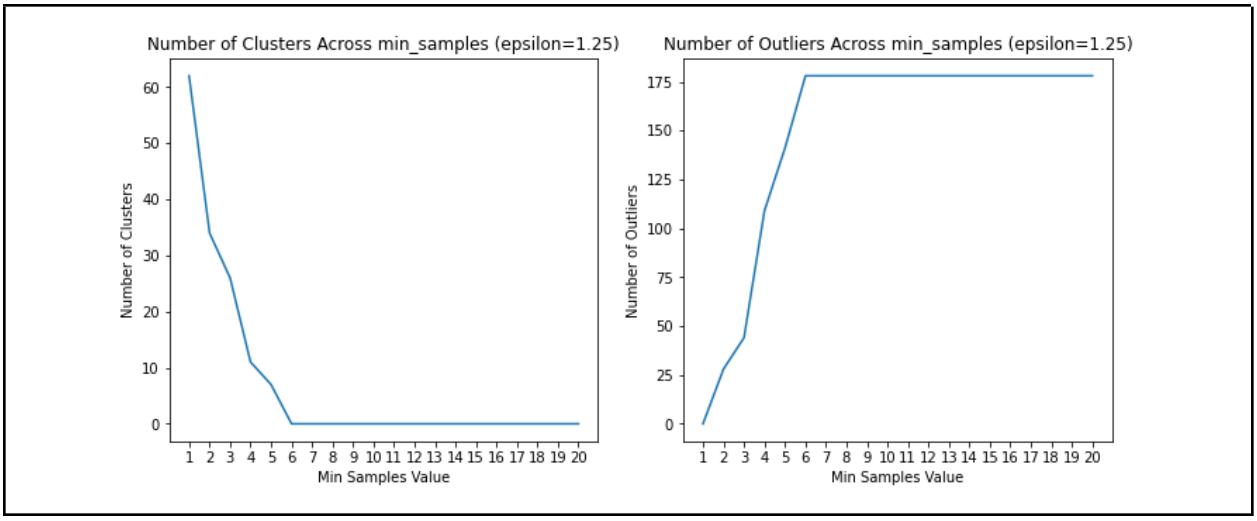
cluster center was 75.66; however, my own calculations yielded a total distance of 104.806. For either calculation, this suggests that each point has a euclidean distance less than 1 from its respective cluster center (on average).

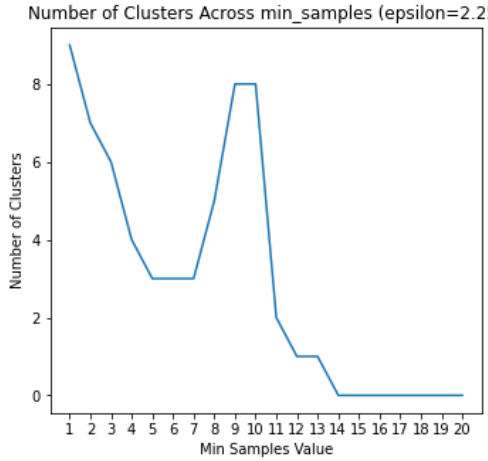
5. **Building on one of the dimensionality reduction methods above that yielded a 2D solution (1-3, your choice), use dBScan to produce a plot that represents each wine as a dot in a 2D space in the color of its cluster. Make sure to suitably pick the radius of the perimeter ("epsilon") and the minimal number of points within the perimeter to form a cluster ("minPoints") and comment on your choice of these two hyperparameters.**
    a. For this question, I used my results from the t-SNE transformed data with perplexity = 20. To run dBScan, I used the DBSCAN object from sklearn.cluster. I first tested different epsilon values by setting min_samples fixed at 2 and iterating through values of epsilon between 0.75 and 4.5 in increments of 0.25. For each epsilon value, I found the average silhouette score of the clustered data and plotted these values across values of epsilon to see which values were optimal. I found four optimal values to test (1.25, 2.25, 3.5, and 4.0). For each of these, I then iterated through possible values of min_samples between 2 and 20. I found the number of clusters yielded by each of these combinations as well as the number of outliers, and again plotted these trends for analysis. Lastly, I took the two values of epsilon that yielded the most promising results from my min_samples analysis (i.e. the most intuitive number of clusters and the fewest outliers) and tested the values of min_samples that appeared to produce few outliers and a reasonable number of clusters. For each of these tests, I produced a scatter plot of the t-SNE data labeled with the dBScan-predicted cluster labels. I then visually inspected the fit of these results to determine the best possible values of epsilon and min_samples.
    b. I first chose epsilon values by setting min_samples constant at 2 as this value was low enough that I didn't run into errors testing epsilon values. Higher values of min_samples sometimes yielded data that only had a single cluster label - which isn't usable for silhouette score. I found the silhouette score for each epsilon value because a higher value would indicate the data was more ideally clustered on average - which would indicate a good value of epsilon. After picking values of epsilon with high average silhouette scores, I iterated through values of min_samples to see which combination of hyper parameters was ideal in terms of the number of clusters produced and the number of outliers. I intuited a reasonable number of clusters would be around 3 with few outliers by visually inspecting the scatter plot shown in question 2. However, I then took combinations of hyper parameters that appeared to yield promising results and

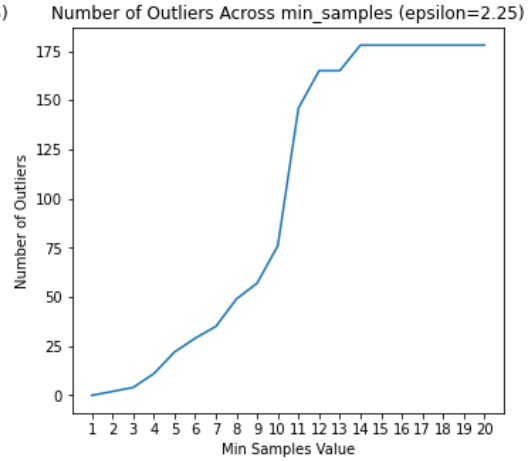
visually inspected the fit using a scatter plot of the clustered data to see which dBScan had the best results.

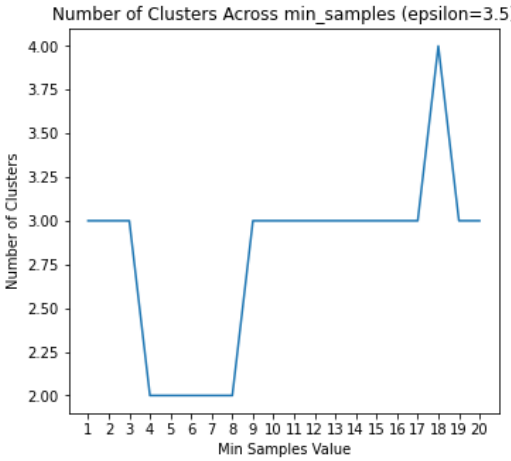
c. As shown in the first plot below of average silhouette score across tested epsilon values, epsilon values of 1.25, 2.25, 3.5, and 4.0 all had silhouette score values above 0.2 - with the silhouette score plateauing at higher values of epsilon. The table beneath this plot shows the number of clusters yielded by varying min_samples across these four values of epsilon - as well as the number of outliers produced by each dBScan. The epsilon values 3.5 and 4.0 consistently produced between 2 and 4 clusters for all tested values of min_samples with fewer than 60 outliers for any one combination of parameters. Additionally, the epsilon value 4.0 had a maximum number of outliers of about 18, indicating that this epsilon value may be ideal. The scatter plots below confirm this. The results of epsilon = 3.5 and min_samples = 9 as well as epsilon = 4 and min_samples = 10 both show three reasonably labeled clusters with a group of outliers at the bottom-left of the plot. However, in the epsilon = 3.5 plot, we observe that there is one point in the red cluster labeled as green - which is not found in the epsilon = 4 solution.

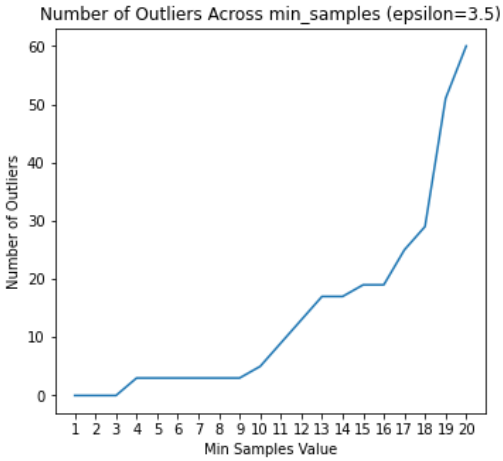


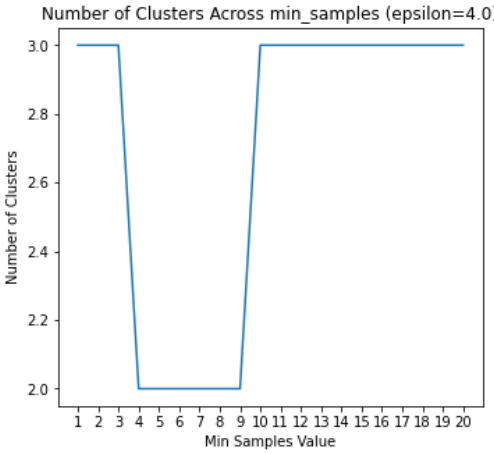


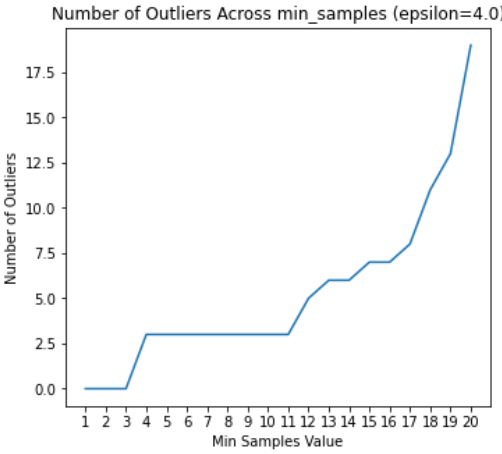

Number of Clusters Across min_samples (epsilon=2.25)
Number of Clusters
Min Samples Value

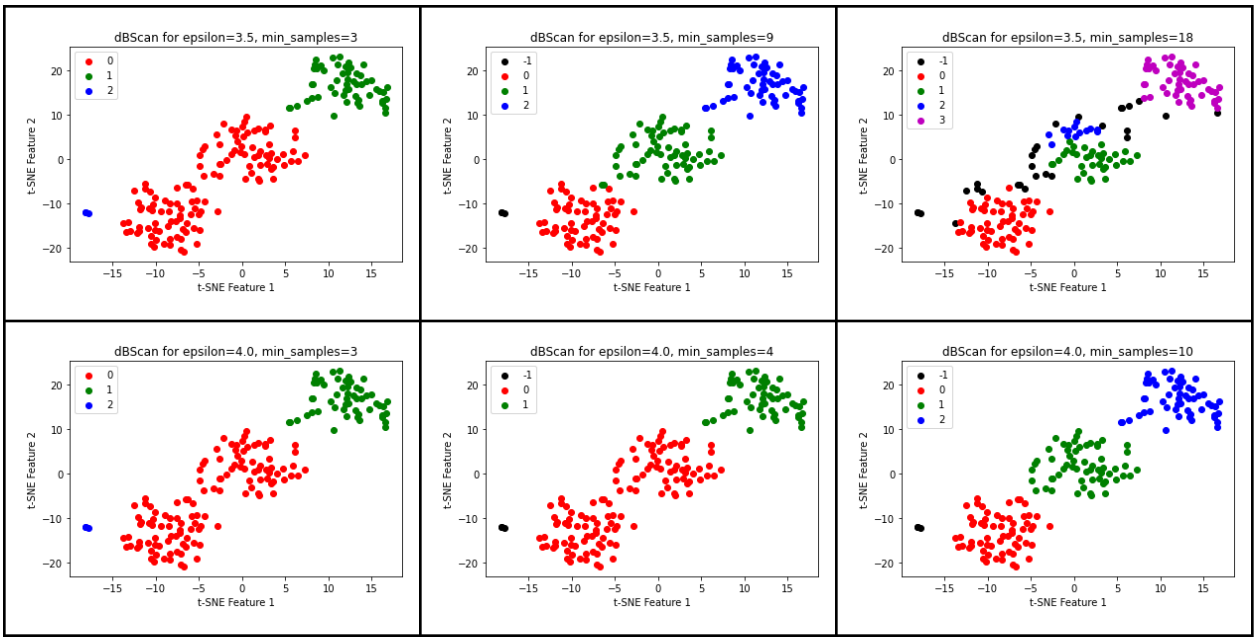

Number of Outliers Across min_samples (epsilon=2.25)
Number of Outliers
Min Samples Value


Number of Clusters Across min_samples (epsilon=3.5)
Number of Clusters
Min Samples Value


Number of Outliers Across min_samples (epsilon=3.5)
Number of Outliers
Min Samples Value


Number of Clusters Across min_samples (epsilon=4.0)
Number of Clusters
Min Samples Value


Number of Outliers Across min_samples (epsilon=4.0)
Number of Outliers
Min Samples Value



d. I found the optimal values of epsilon and min_samples to be 4.0 and 10 respectively. This combination yielded a reasonable number of clusters with few outliers. Additionally, the epsilon value of 4.0 had a higher average silhouette score (approximately 0.3) than other tested values when min_samples was held constant at 2. This indicates that the majority of points were well-classified. Lastly, the scatter plot of this solution shows that the three distinct groups of t-SNE transformed data are appropriately classified as three clusters, with a small group of outliers.

Extra Credit

A. Given your answers to all of these questions taken together, how many different kinds of wine do you think there are and how do they differ?

a. Given my analysis, there seem to be three kinds of wine. This was the optimal number of clusters yielded by my silhouette analysis on PCA-transformed data, as well as the most reasonable clustering yielded by dBScan on t-SNE-transformed data. Intuitively, I suspect that the three kinds of white would be red, white, and rosé. However, the loadings of the PCA conducted in question 1 can provide more insight on the dimensions for which these groups may differ. The first principal component was most highly correlated with total phenols, flavonoids, and OD280, while the second component was most highly correlated with alcohol, color intensity, and proline. Research suggests that red wines have a higher content of total phenols than white wine. For alcohol content, red and white wines have a higher content on average than rosé. Additionally, it's likely that color intensity is also reflective of red vs. rosé vs. white wine. Therefore, it seems likely

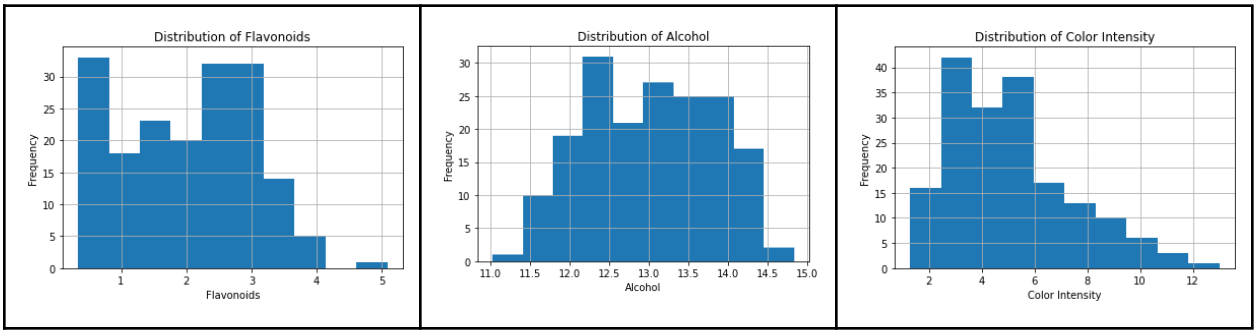
that the three clusters we observe may be representative of red, white, and rosé wines.

B. Is there anything of interest you learned about wines from exploring this dataset with unsupervised machine learning methods that is worth noting and not already covered in the questions above?

a. In order to further explore the dataset, I used the loadings found after running PCA in question 1 to see which features might be important in classifying wines. I found that the first principal component was most highly correlated with 'flavonoids', while the second component was highly correlated with 'color intensity' and 'alcohol.' I then plotted histograms of these three variables using the original, unnormalized dataset to see how these features are distributed in the dataset. I also found the correlations between color intensity and alcohol, flavonoids, and hue.

b. I decided to conduct this analysis because I wanted to see if the underlying distributions of important features from the PCA analysis reflected any of the clustering or separation observed in the rest of my analysis. I used 'flavonoids', 'color intensity' and 'alcohol' as they were the variables most highly correlated with the first two principal components from question 1. This was an ideal choice as the first two principal components explained over half of the total variance in the dataset. I also found the correlations between color intensity and the other features to test my hypothesis of red, white, and rosé wines being representative of the three kinds of clusters observed.

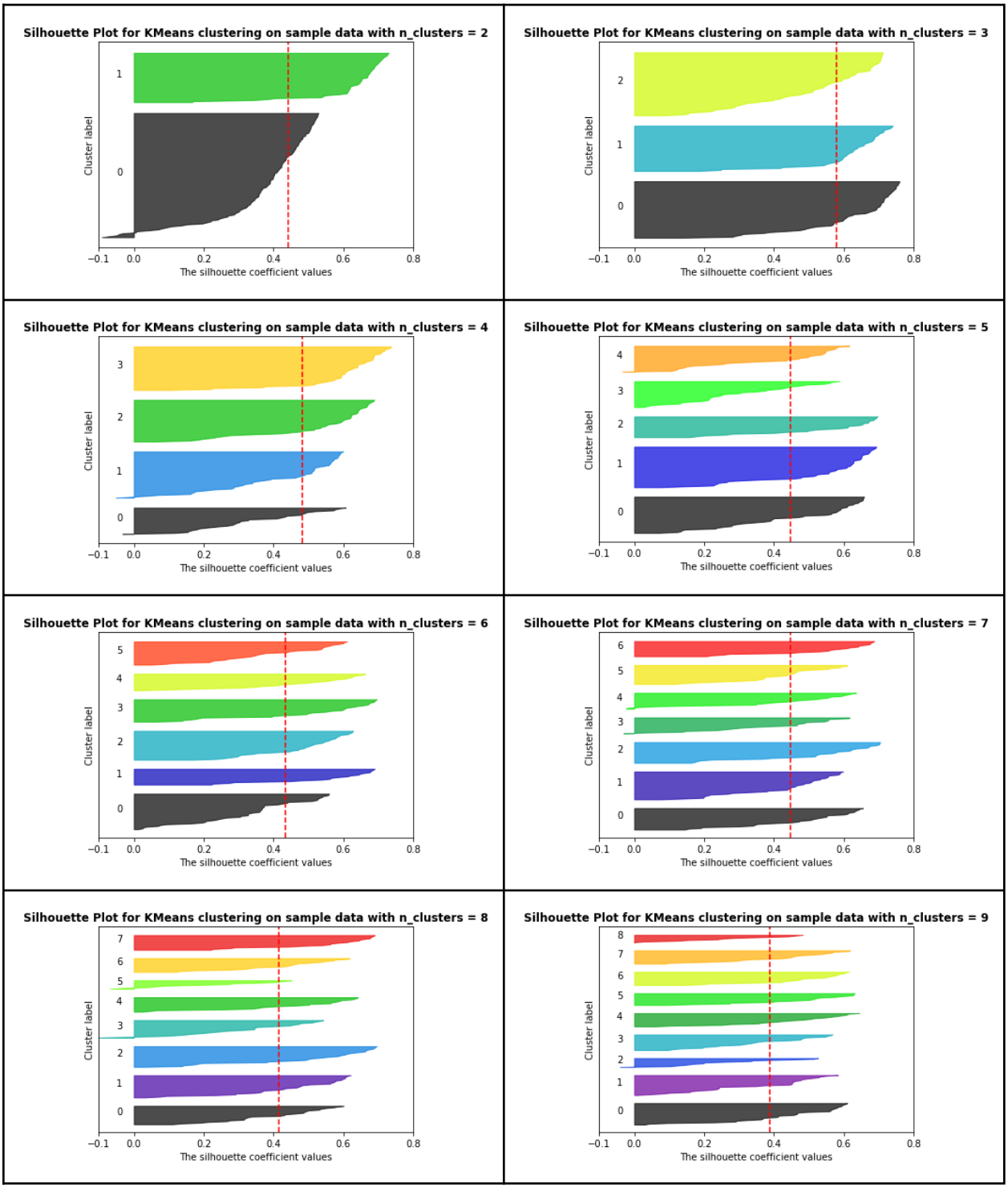
c. Both the distributions of flavonoids and color intensity show some right skew, while the distribution of alcohol appears more symmetric - possibly normally distributed. Interestingly, the distribution of flavonoids appears to be bimodal, which may reflect wines falling into one of two primary categories with respect to this attribute. Additionally, the correlations between color intensity and alcohol, flavonoids, and hue were 0.54636, -0.17238, and -0.52181 respectively.

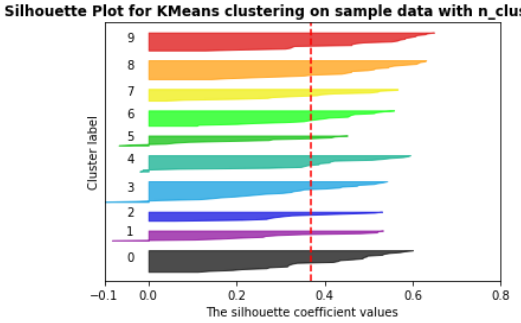


d. My most interesting finding from this analysis is that the distribution of flavonoids appears to be bimodal. In the PCA analysis conducted in question 1, I

found that the first principal component (which explained 36.2% of the total variance in the dataset was most highly correlated with flavonoids. The bimodal nature of this distribution indicates that wines may fall into two main categories in terms of flavonoids, which could have contributed to the clusters we observe in the 2-dimensional solution using PCA-transformed data. Additionally, flavonoids had little correlation with color intensity - which was strongly correlated with the second principal component. This makes sense as the two components are orthogonal to one another, and therefore uncorrelated.

Appendix


Silhouette Plot for KMeans clustering on sample data with n_clusters = 2
Silhouette Plot for KMeans clustering on sample data with n_clusters = 3
Silhouette Plot for KMeans clustering on sample data with n_clusters = 4
Silhouette Plot for KMeans clustering on sample data with n_clusters = 5
Silhouette Plot for KMeans clustering on sample data with n_clusters = 6
Silhouette Plot for KMeans clustering on sample data with n_clusters = 7
Silhouette Plot for KMeans clustering on sample data with n_clusters = 8
Silhouette Plot for KMeans clustering on sample data with n_clusters = 9
Cluster label
The silhouette coefficient values
−0.1
0.0
0.2
0.4
0.6
0.8


Silhouette Plot for KMeans clustering on sample data with n_clusters = 10
Cluster label
9
8
7
6
5
4
3
2
1
0
−0.1
0.0
0.2
0.4
0.6
0.8
The silhouette coefficient values