FML Homework 5 - Erin Choi

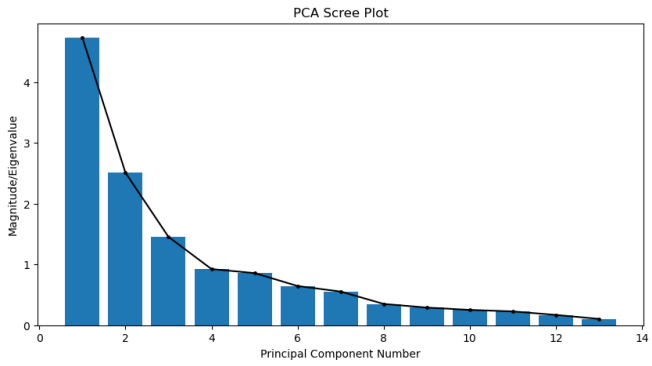
Data Preprocessing

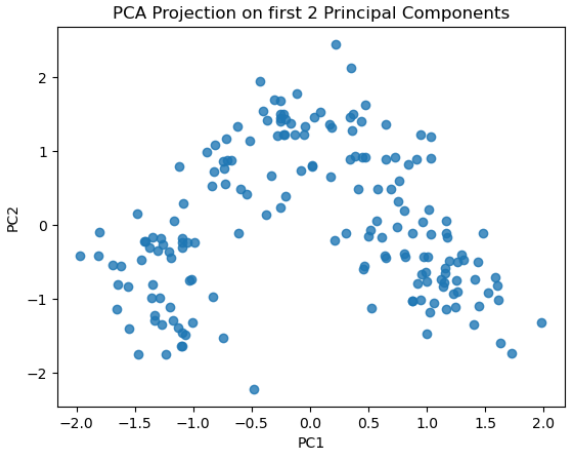
I set a seed of 42 to help make results reproducible. After loading the data, I observed the distributions of the features using histograms and boxplots. I used StandardScaler to standardize the data and get all features in a similar range.

Question 1

I fitted a PCA to my scaled data then transformed the data. I obtained the Eigenvalues using pca.explained_variance_ and created a scree plot to visualize them. I then plotted the first two principal components (PCs) to display the 2D solution of the PCA. I used pca.explained_variance_ratio to get the percentage of variance explained by each PC and added the first two values together to see how much of the variance is explained by the 2D projection. To interpret the dimensions, I viewed how important each feature is to PC1 and PC2 by sorting the absolute values of the Eigenvectors (pca.components_) for both in decreasing order, as the Eigenvectors act similarly to feature importances here.

Referring to the Data Preprocessing section, it is important to standardize data before performing PCA as it maximizes variance; if different features have different variances, there may be biases to certain features, automatically making those features more important when creating PCs. The attributes of the fitted PCA contain important information about the PCA including Eigenvalues and proportions of variance explained by PCs, so I could easily manipulate attribute values to answer the questions.





The scree plot above on the left shows that **there are 3 Eigenvalues that are greater than 1**: 4.732 (PC1), 2.511 (PC2), and 1.454 (PC3). The projection on the first two PCs is also shown above to the right. Adding together the proportions of variance explained by PC1 (0.362) and PC2 (0.192) results in a total of **55.4% of the total variance that is explained by the two dimensions.**

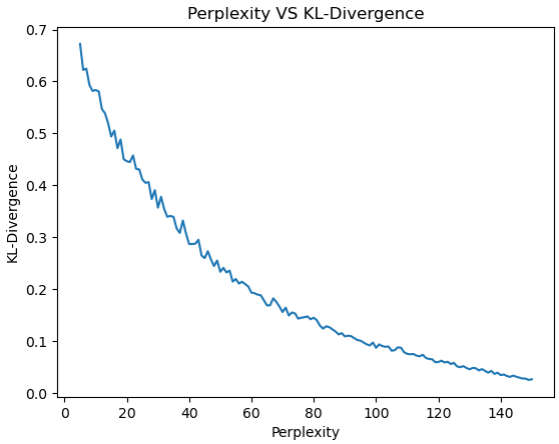
Since there are 3 Eigenvalues above 1, the first 3 PCs are important for representing the dataset. As aforementioned, the 2D projection only explains about 55% of the variance, and adding in the variance explained by PC3 (0.111) would bump the total variance explained up to about 66%. The graph of the 2D projection shows, vaguely, about 3 clusters with some outliers; with a 3D projection, we would expect the

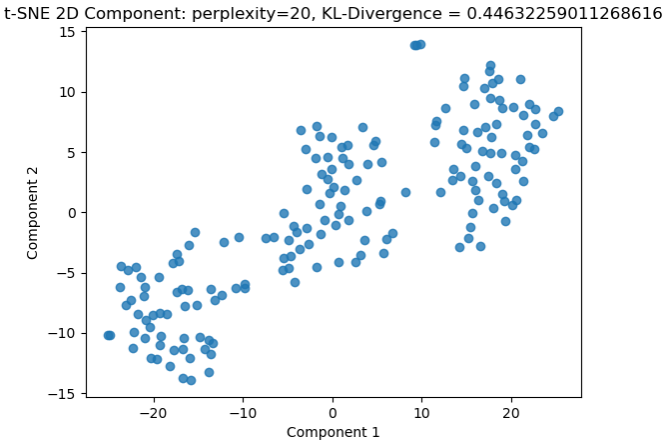
clusters to be more clearly separated since the third PC is also somewhat important. The most important features (largest Eigenvectors, above 0.3) for PC1 are Flavonoids, Total_Phenols, OD280, and Proanthocyanidins. The most important attributes for PC2 are Color_Intensity, Alcohol, Proline, and Ash. Thus **PC1 can be interpreted as some measure of flavors and nutrient content, and PC2 can be interpreted as some measure of color and acidity.**

Question 2

For each integer value from 5 to 150, I used t-SNE on the scaled data with that value of Perplexity, saving the resulting KL-Divergence (from tsne.kl_divergence_) for each t-SNE in a list, then plotted the KL-Divergence for each Perplexity value. I also plotted the first two components of t-SNE with a Perplexity of 20.

Like PCA, sklearn's TSNE has attributes including kl_divergence_ that can be used to answer the questions. The first two components were plotted to display the 2D visualization of the t-SNE solution.





The plot of KL-Divergence for each Perplexity value from 5 to 150 is shown above on the left. The 2D component for t-SNE with Perplexity=20 is shown above on the right. The KL-Divergence of this embedding is 0.4463.

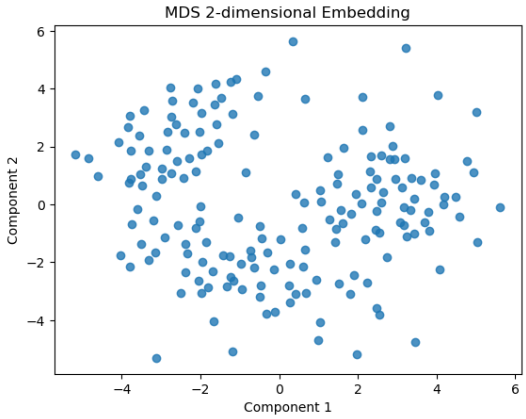
The plot of Perplexity versus KL-Divergence shows that **as Perplexity increases, KL-Divergence decreases.** Perplexity represents the number of neighbors that are considered, so with more neighbors considered for each point, the probability distributions of the original space and the lower-dimension space become more similar, with gaps between clusters in the embedding becoming larger. This makes sense since, with more neighbors considered, more information about the data's global structure is retained in the low-dimensional embedding. Like the PCA solution, the t-SNE 2D component with a Perplexity of 20 shows **3 clusters, but the clusters are more clearly separated in the t-SNE solution.** t-SNE is known to sacrifice global structure for local structure, so while the clusters are better separated, its global structure is likely not as meaningful as the local structure.

Question 3

I used MDS on the scaled data with n_init of 100 and max_iter of 10000, then plotted the resulting 2D embedding (the first two components of the MDS solution). I found the stress of the embedding using the mds.stress_ attribute and normalized this value by dividing it by the sum of squared distances, obtained from mds.dissimilarity_matrix_.

I increased the n_init value for the MDS to increase the number of times the algorithm is run with different initializations (as per the MDS documentation), giving the model more chances to find better outputs (with lower stress). I also increased the max_iter to allow the algorithm to do more iterations and potentially find better outputs. The stress was normalized in an attempt to make the value more interpretable and telling of how well the embedding fits the data.

The resulting stress of the 2D embedding is 21105.55, with the calculated normalized stress being 0.1365. The visualization of the 2D embedding is shown to the right.

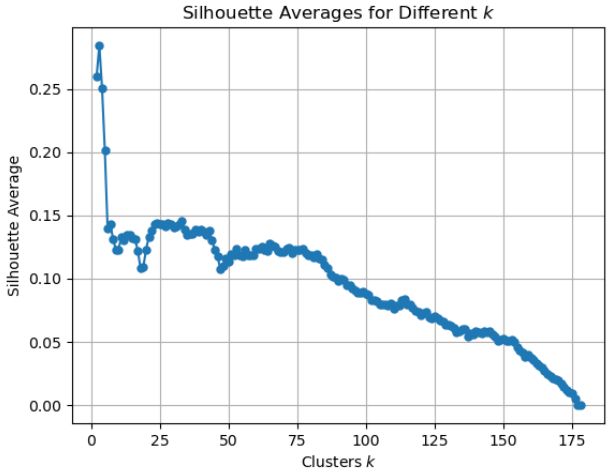


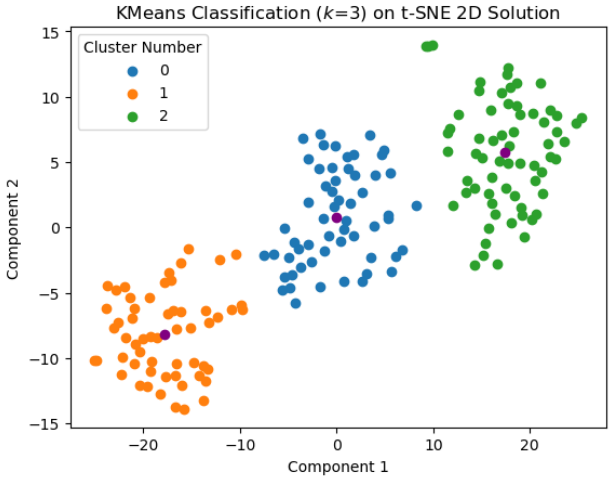
The raw stress value of the embedding is quite large; this makes sense since it is difficult for MDS to preserve the structure of the data when reducing 13 features to 2 dimensions. When normalized to be between 0 and 1, with 0 representing a perfect fit, the value of 0.1365 indicates a fair preservation of distances in this embedding. In the plot of the MDS 2D embedding, 3 vague clusters can again be identified along with some outliers. **When compared to the plot of the t-SNE solution, the t-SNE graph has much clearer separation between the 3 clusters.** This is as expected - we know t-SNE preserves local structure better than MDS as it aims to preserve data points' nearest neighbors rather than comparing every pair of points.

Question 4

I chose to build on the 2D solution of t-SNE with a Perplexity of 20. I implemented the Silhouette method by performing k-Means clustering with 2 to 178 clusters, plotting the average silhouette score for each k-Means run, and identifying the peak of the plot as the k-Means run with the number of clusters that yields the highest silhouette score. I then ran k-Means with that *k* value to predict labels for the points in the t-SNE 2D solution and plotted the labeled/colored points along with the cluster centers (from kmeans.cluster_centers_). I found the sum of the distances of all points to their cluster centers by summing the minimum values among each point's distance from each cluster center (found using kmeans.transform()).

I selected the t-SNE solution to build on because its 2D embedding had the best separation between clusters. The components/dimensions of t-SNE are not very interpretable anyways, so for the purposes of this assignment, it was fine that the t-SNE solution only had a good local structure and not a great global structure. I did clustering for 2 to 178 clusters since there is no point in labeling all points as part of one cluster, and there are 178 points in the data, which is the maximum number of possible clusters for the data.





The silhouette method plot is shown above to the left. The peak average silhouette score occurred at the second point on the plot. Since k-Means was done beginning with 2 clusters, **the optimal number of clusters *k* is 3.** The k-Means classification labels on the t-SNE 2D solution are shown above to the right along with each cluster center in purple. **The total sum of distances of all points to their respective cluster centers was found to be 861.648.**

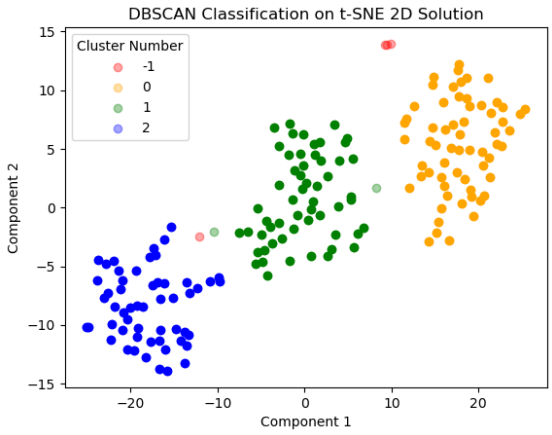
The plot that shows each wine as a point in the t-SNE 2D embedding displays 3 different colored clusters. It labeled the points as expected when visually inspecting the t-SNE solution, with cluster centers also appearing as expected, right in the middle of each spherical cluster. Dividing the total sum of distances of points to their cluster centers by the number of clusters produces an average distance of each point from its cluster center of 4.84. This value could be informative for hyperparameter tuning the epsilon value in Question 5.

Question 5

Again, I chose to build on the 2D solution of t-SNE with a Perplexity of 20. I ran DBSCAN multiple times, varying epsilon from 0.5 to 4 and varying min_samples from 1 to 5. For each run, I plotted the labeled points to observe visually how the hyperparameter settings affected clustering.

As described in Question 4, I built on the t-SNE solution because its 2D embedding had the best separation between clusters. I varied the hyperparameters starting from low numbers until I passed what I considered the best values to see what happens when the values are either too small or too large.

After visualizing the DBSCAN clustering with multiple combinations of epsilon and min_samples values, **I chose the optimal values to be 3 for epsilon and 4 for min_samples** based on how many clusters were formed and what points were or weren't included in the clusters. The plot of the DBSCAN clustering on the t-SNE solution is shown to the right. The more saturated points are the core points of their clusters, while more transparent points are not.

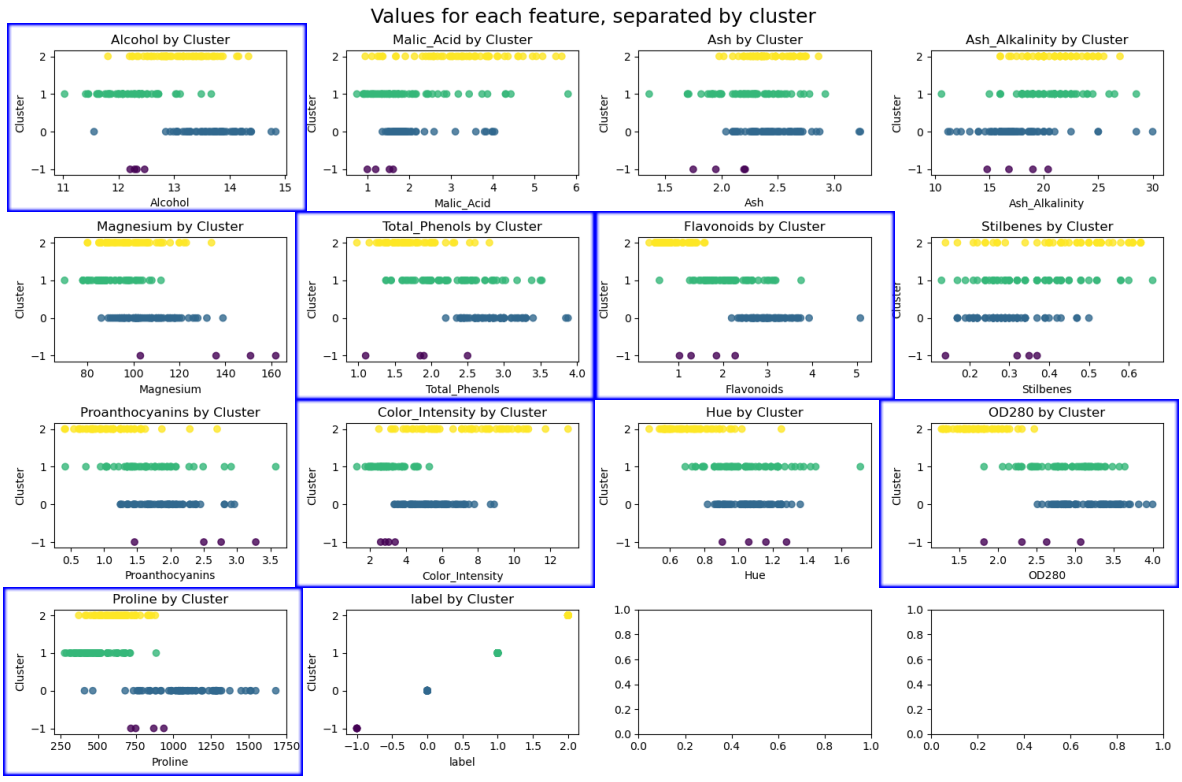


The DBSCAN clustering with epsilon=3 and min_samples=4 yielded 3 clusters with several outliers/noise points labeled as -1. The clustering seems robust with these hyperparameters, as almost all labeled points (not outliers) are core points of their clusters; it appears only 2 points in the green cluster are non-core points. As I tried different hyperparameter values, I noticed that an epsilon greater than 3 clustered almost all the points together, while an epsilon value less than 3 forms too many clusters (i.e. more than 3 clusters, which is obviously the best number of clusters by visual inspection). Also, with min_samples greater than 4, many more points are labeled as outliers, while a min_samples value less than 3 produces fewer clusters, with some outliers (the blue points at the top right of the graph) forming their own cluster. Thus I determined that 3 and 4 are the best values for epsilon and min_samples, respectively. DBSCAN provides a better solution than k-Means because it accounts for outliers, labeling them as -1 instead of making them their own cluster when using the optimal hyperparameters.

Extra Credit Part A

In addition to looking at the resulting 2D spaces/embedding solutions and clustering plots above, I plotted the unscaled values for each wine attribute separated based on labels from DBSCAN. I did this to view how the distribution of values for each attribute differs depending on the cluster label.The plots for values for each feature, separated by DBSCAN label, are shown below.

Based on the results of all questions above, I believe there are **3 kinds of wine**, at least in this dataset. **The attributes for which value distributions differ the most between the 3 clusters appear to be Alcohol, Total Phenols, Flavonoids, Color Intensity, OD280, and Proline** (boxed below in blue). Interestingly, these are features that are among the most important to PC1 and PC2 when PCA was done on the data in Question 1. It seems dimension reduction by t-SNE was done based on similar important features as in PCA, and the differences in these features worked as the distinguishing characteristics of clusters in DBSCAN on top of the t-SNE solution.



Overall, wines in Cluster 0 (blue) are among those with the highest average alcohol content, total phenols, flavonoids, and proline content. Wines in Cluster 1 (green) are among those with the lowest average alcohol content, color intensity, and proline content. Cluster 2 wines (yellow) are among those with the lowest total phenols and flavonoids as well as the highest (but most varying) average color intensity.

Through some quick searches, I found that red wines have the highest alcohol, phenol, and flavonoid content, while white wines have lower proline, phenol, and flavonoid content. Sparkling wines generally have lower alcohol content, and intuitively, red wines are more intense in color than white wines. Wines in Cluster 0 seem to have the closest qualities to red wines, with generally high values for these distinguishing features. Wines in Cluster 1 seem to be similar to white wines, especially with the lowest average color intensity. Wines in Cluster 2 could be sparkling/rose or generally "other" wines since their color intensities can vary a lot from white to red. I find it impressive that I could infer this much from clustering on a 2D embedding.

Extra Credit Part B

Looking at the plots in Part A, the attributes of the outlier wines (purple) generally appear to fit best into the distributions of Cluster 1 (green). Based on my idea that the DBSCAN clusters could potentially map to red/white/other wines, these noise points could be unique kinds of white wines.

The 2D t-SNE embedding that DBSCAN was used on (graph in Question 5) has clusters appearing in a general upward trend. Though t-SNE is known not to preserve global structure well, I thought it was interesting that Cluster 0 (potential "red wine" cluster) was the top right cluster with the highest component values, Cluster 1 (potential "white wine" cluster) was the intermediate cluster, and Cluster 2 (potential "sparkling, rose, or other wine" cluster) was the one with the lowest component values. There could be some global structure preserved by t-SNE in this case, meaning red wines may be the "strongest" wines in terms of many attributes, white wines could be middling in many attributes, and sparkling/other wines may be the "weakest" overall.

Finally, looking again at the plots in Part A, the distributions of values for malic acid concentration, ash content, and stilbenes appear to be very similar or entirely overlapping between the 3 clusters. Malic acid is an acid found in grapes; a wine will taste flat if there is not enough of it or sour if there is too much. It makes sense that this value is similar and overlapping between clusters because all the wines must have a good balance of malic acid concentration for them to taste just right. Next, ash content is mostly minerals from soil absorbed by grapes; if ash content is similar between all clusters, the concentrations of minerals contained in different types of wines are similar. This leads me to guess that grapes grown specifically for wines go through similar processes irrespective of wine type, which causes them to absorb a somewhat specific amount of minerals. Lastly, stilbenes are a specific kind of chemical compound, and humans get them mostly from consuming grapes or wine. The similar amounts of stilbenes in the 3 types of wine could just mean they are, indeed, all made from grapes.