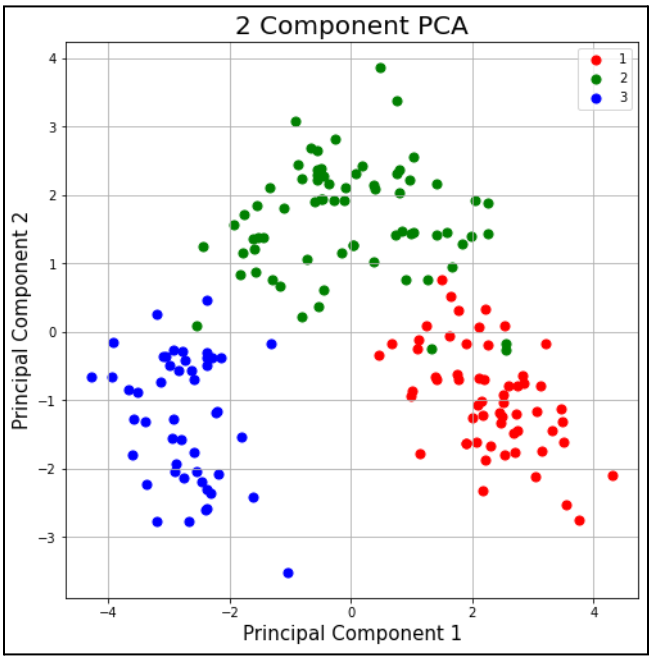
LAB 9 REPORT

[redacted]

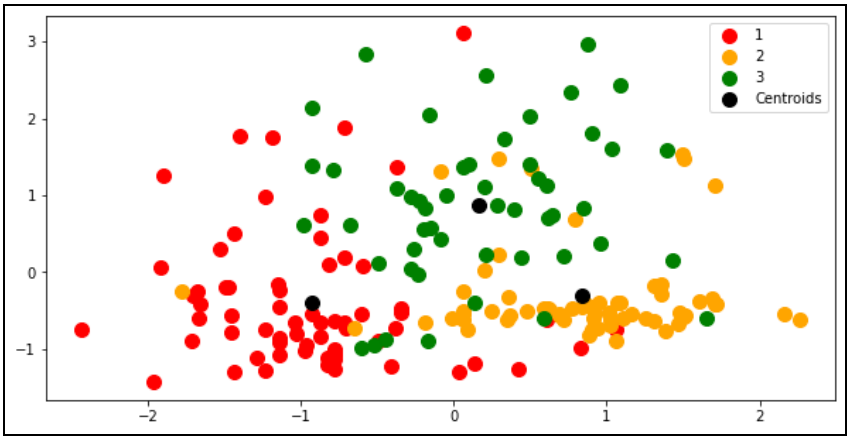
Q1)

For the first question, I preprocessed the data, there were 0 Nan values in data, all features in data were continuous so there was no need of encoding the features. The only thing to be done in preprocessing was normalizing all features on the same scale, this I had done with the help of Standard Scaler from sklearn library.

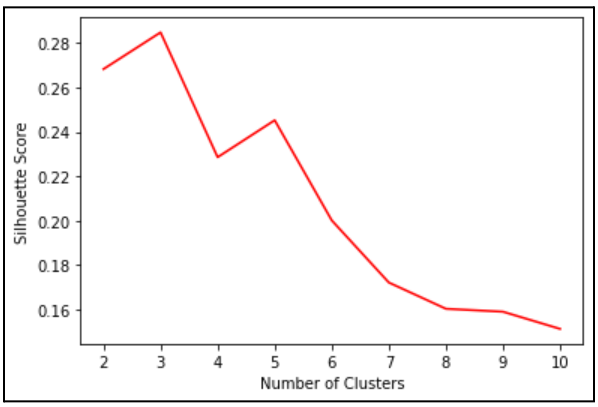
For Dimension Reduction I used PCA and took the number of principal components equal to 2. After doing data visualization on a scatter plot it was visible that the data was already clustered in 3 in the graph.



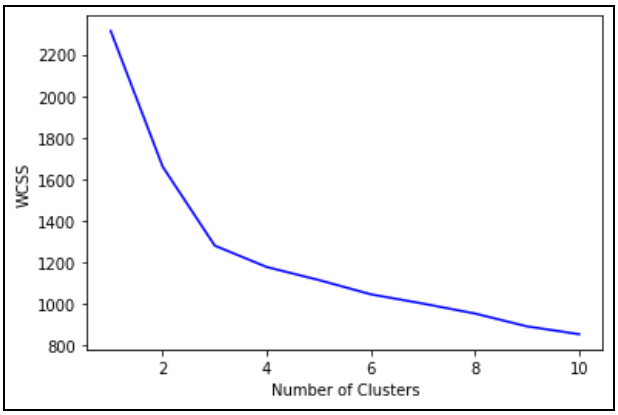
After this I implemented the K-means clustering algorithm from sklearn while taking the value of number of clusters =3 and then the same on a scatter plot.



Next, I used The k-means clustering algorithm for different values of the number of clusters from 2 to 11 and calculated there silhouette score. And plotted the performance on the graph, from the graph it is visible that the silhouette score is highest(greater silhouette being better) for number of clusters = 3.



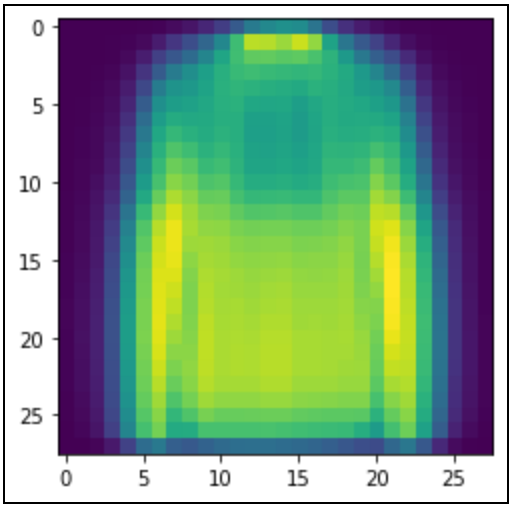
For the last part, the elbow method was used in which for different values of k (here 1 to 11) , we calculate the WCSS(within cluster sum of squares) and plot them on a graph. By looking at the graph, the value of k is chosen where the graph looks like an elbow joint. On using the elbow method on data the optimal value found was equal to 3, which is in line with our previous findings.



Q2)

For the second question, I built the K-means algorithm from scratch. So, first, I chose random numbers within a range of row numbers to use as data point indexes. I used a for loop for a particular number of iterations after using these randomly chosen data points as initial centers. Because the dataset is too huge and maintaining a high value takes too long, the number of iterations was kept low. The kmeans function provides a dictionary containing all separated(classified)clusters as well as centers. The length of each cluster can be determined by examining the length of the array corresponding to ith cluster in the dictionary.

For visualizing images, I used the imshow() function from matplotlib. For a datapoint(that corresponds to an image) we have 784 pixel values, which is basically a 28x28 pixel image. So I reshaped the array into 28x28 array and put it in the imshow() function and it plotted images as shown below

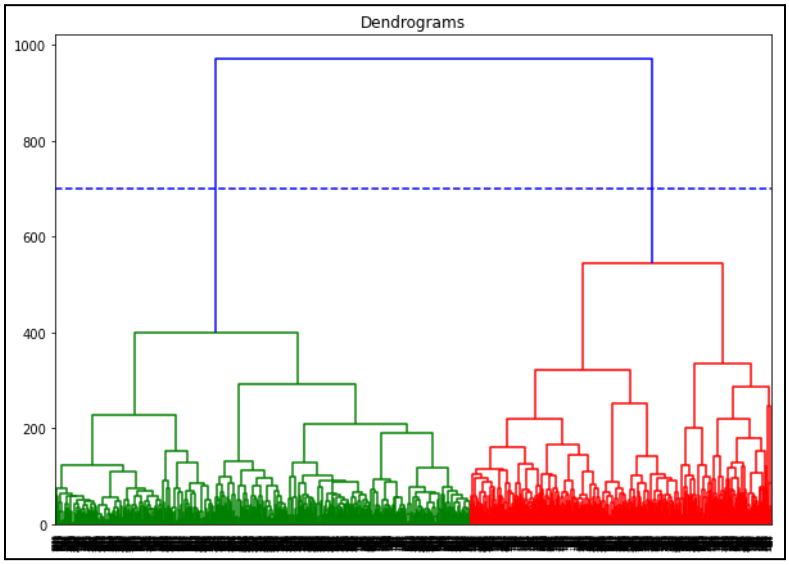


For the next few parts, I created a different function, kmeans_b with the only change being that unlike the previous one it was not initialized with random values but each initial center is a datapoint from each cluster found previously. Then I did the same previous things: Checked number of points in each cluster, plotting centers, and 10 images of each cluster. For calculating the SSE scores for the two methods, I defined a function calculate_sse which takes the clusters and centers and k as parameters and in the function I calculated the distance of a cluster's data points from their respective centers and added them all up for 10 clusters. On calculating sse it was found that the method with initialization with random value had a larger SSE than the other method.

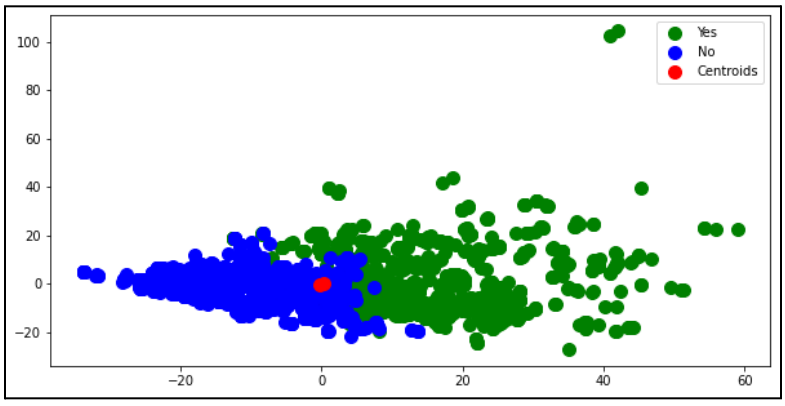
Q3)

For the third question, for using the data I imported all the files on drive. Then I created 2 lists (one for yes and other for no) to store the images using os.listdir function. After this I converted the images into array by using the Image.open() function and reshaped the image into (28*28 array) after this I flattened the array.

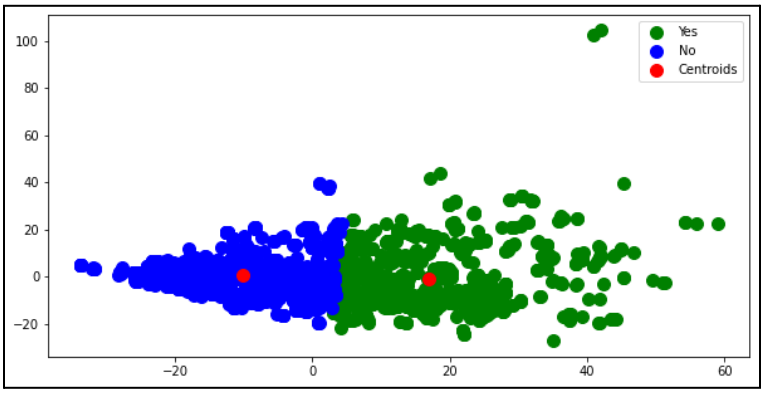
As the data was created from images there was no need of dropping Nan values (as will be 0 NaN values) and doing encoding (all features are continuous). The only thing that was to be done in preprocessing was normalizing on the same scale using the StandardScaler function.
Then I did dimension reduction by using PCA and reduced the number of features to 400.
In Agglomerative hierarchical clustering, I plotted a Dendrogram as shown below:



The Agglomerative hierarchical clustering gave visualization of clusters as shown below:



While the kmeans(done using sklearn same as Q.1) clustering gave visualization of clusters as shown below



On comparing the results from Kmeans clustering and Agglomerative hierarchical clustering it is visible that the graph for both is mostly the same except the centroids in Agglomerative hierarchical clustering graph is close to each other while centroids in K-means clustering graph is close to each other