Dimensionality Reduction & SVMs

Ankit Bhadu
2018CSB1073
Indian Institute of Technology
Ropar

Abstract

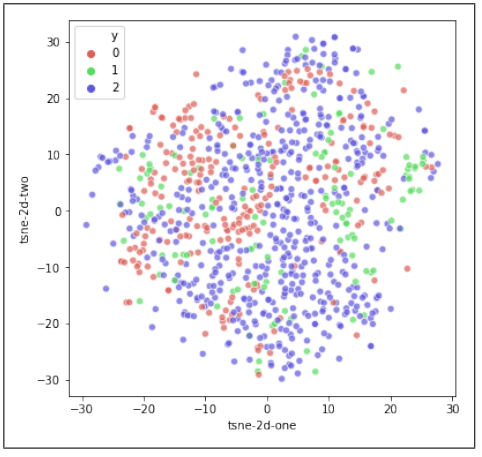
In this document we perform Dimensionality Reduction and Visualization with PCA, LDA and tSNE, look at data classification with linear and non-linear SVM and perform face recognition using Eigenfaces.

1 Introduction

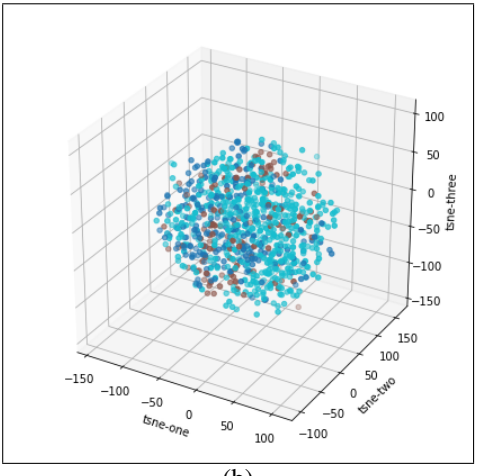
This document is divided into three sections:
1) Principal Component Analysis and Eigenfaces for Face Recognition
2) Dimensionality Reduction and Visualization with PCA, LDA and tSNE
3) Data Classification with Linear and Non-linear SVMs

1.1 Principal Component Analysis and Eigenfaces for Face Recognition

1. **Performing PCA** :
   PCA is a technique for reducing the number of dimensions in a dataset without losing important information. It uses the correlation between some dimensions and tries to provide a minimum number of variables that keeps the maximum amount of variation or information about how the original data is distributed. Here we perform PCA with **no of components being 100**
2. **Visualization using tSNE** :
   After performing PCA with no of components 100, we now have a 100D vector. We now plot a scatter plot via projecting these points into 2D and 3D using tSNE. We choose the first three personalities for this purpose.
   t-SNE differs from PCA by preserving only small pairwise distances or local similarities whereas PCA is concerned with preserving large pairwise distances to maximize variance.
   Observation: However no clear clusters were observed in the 2D and 3D plots because when a 100D data was converted to 2D(which was already dimensinally reduced from approx 3000) the unique features which could differentiate among faces were lost while reducing dimensions. Since t-SNE is not a clustering algorithm but a dimensionality reduction algorithm therefore we cannot always make any inference based only on the output of t-SNE. This is because it maps the multi-dimensional data to a lower dimensional space, the input features are no longer identifiable. So essentially it is mainly a data exploration and visualization technique and hence we should not infer classifiability of data on the basis of these plots.
3. **Classification using KNN**
   Now we classify the test faces by assigning them the label of the nearest neighbour.



(a)



(b)

Figure 1: Visualization through tSNE for first three personalities

Total explained variance by principal components: 0.8079198598861694

	precision	recall	f1-score	support
0	0.66	0.73	0.70	78
1	0.56	0.66	0.60	38
2	0.75	0.73	0.74	159
3	0.57	0.27	0.36	30
4	0.50	0.57	0.53	37
accuracy			0.66	342
macro avg	0.61	0.59	0.59	342
weighted avg	0.67	0.66	0.66	342

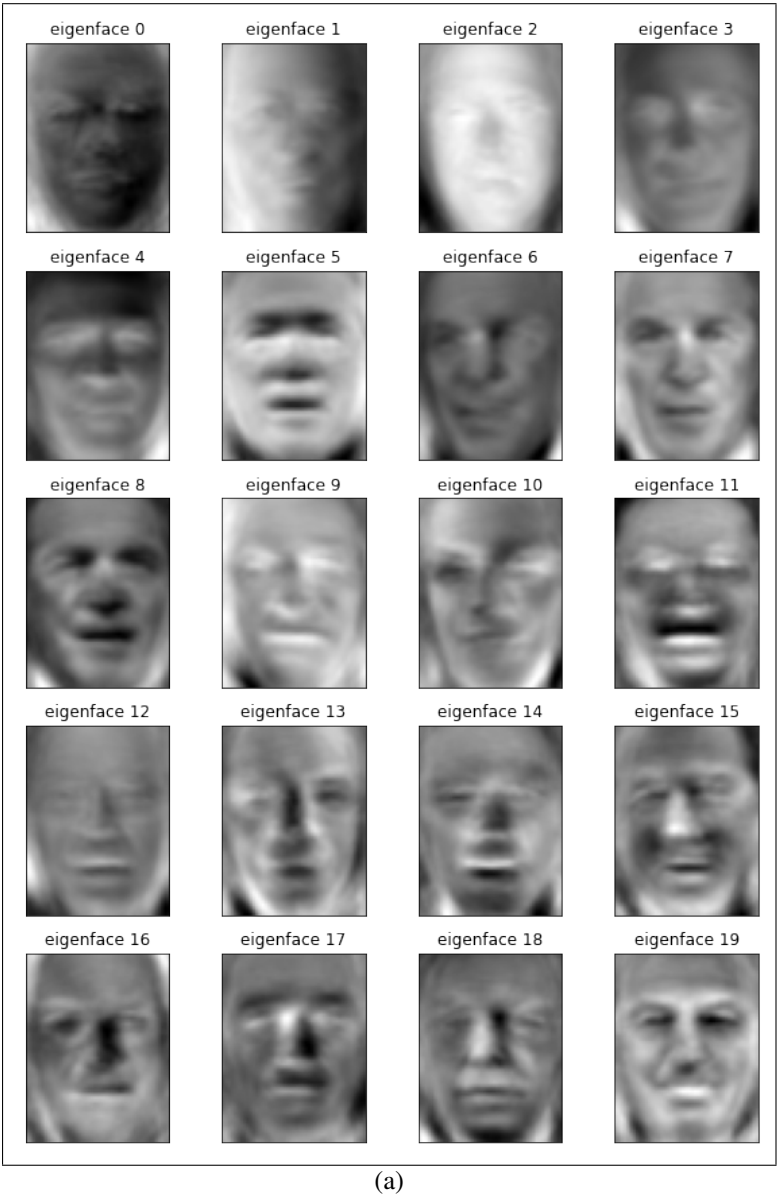
(a)

Figure 2: Classification report

We have used the minkowski metric. Classification report is shown in figure 2. We could classify with an accuracy of 66% among the 5 classes.

4. **Eigen Faces**:
   Eigen Faces basically means a set of eigen vectors in Facial recognition problems. The eigenvectors are derived from the covariance matrix of the probability distribution over the high-dimensional vector space of face images. Using this approach, we can take high-dimensional data and reduce it down to a lower dimension by selecting the largest eigenvectors of the covariance matrix and projecting onto those eigenvectors. The first 20 eigenfaces have been presented in figure 3.
   We can **observe** that first few eigen faces the most essential or the most generic features and as the number grows the specificity of the features rises. For e.g. eigen face 19 specifically resembles the face of an old man and eigenface 0 just resembles the basic shape of a face.

5. **Retaining 80% variance and comparing results:**
   Starting with 100 components in PCA, by gradually reducing the no of components we observe that atleast 32 components are required to explain atleast 80% variance in the original training data. At 100 components explained variance was 92% and accuracy with nearest neighbour classification was 68%. Accuracy reduced to 66% when no of components was set to 32. This shows how reducing the number of n_components reduces accuracy as many distinguishing features are lost when reducing dimension.



(a)

Figure 3: First 20 Eigenfaces

Total explained variance by principal components: 0.8079198598861694

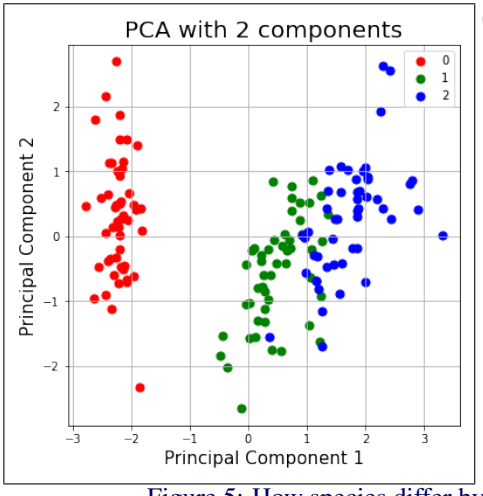
	precision	recall	f1-score	support
0	0.66	0.73	0.70	78
1	0.56	0.66	0.60	38
2	0.75	0.73	0.74	159
3	0.57	0.27	0.36	30
4	0.50	0.57	0.53	37
accuracy			0.66	342
macro avg	0.61	0.59	0.59	342
weighted avg	0.67	0.66	0.66	342

(a)

	precision	recall	f1-score	support
0	0.78	0.64	0.70	78
1	0.58	0.55	0.57	38
2	0.75	0.81	0.78	159
3	0.54	0.47	0.50	30
4	0.40	0.49	0.44	37
accuracy			0.68	342
macro avg	0.61	0.59	0.60	342
weighted avg	0.68	0.68	0.68	342

(b)

Figure 4: Classification report with 32 components(a) vs 100 components(b)



(a)

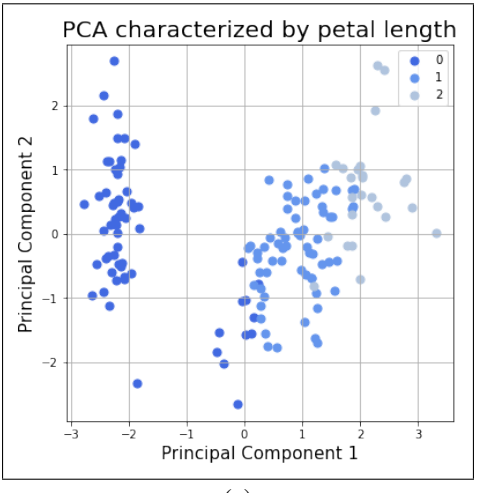
Figure 5: How species differ by their PCA component values.

1.2 Dimensionality Reduction and Visualization with PCA, LDA and tSNE

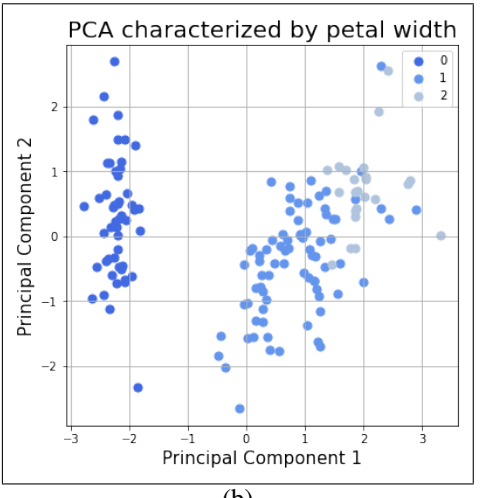
1. **Original features vs Eigen vectors**
   Here we visualize how change along an original feature is affecting values along an eigen vector principal component. First we reduce dimension of **Fisher Iris dataset** from 4 to 2 and visualize it through figure 5. Next we categorize original component values into three categories (0,1,2) with 0 representing the lowest values and 2 representing the high values. Now we can visualize how this power values vary along eigen vectors through figure 6.This way we can confirm our observation from figure 5 that how species vary along PCA components because for eq high petal length corresponds to species 2 and low petal length corresponds to species 0 where as its opposite in the case of sepal width.

2. **LDA and visualization :**
   Through LDA we can project a n class data to n-1 dimensions. Here we first select 2 species at a time and plot their projection on 1D i.e. a straight line using LDA. Then taking all three classes we can plot in 2D using LDA. LDA maximizes inter-class scatter while minimizing intra-class scatter. Lines of separation have been plotted in each plot using SVM. Refer figure 7 for these plots.

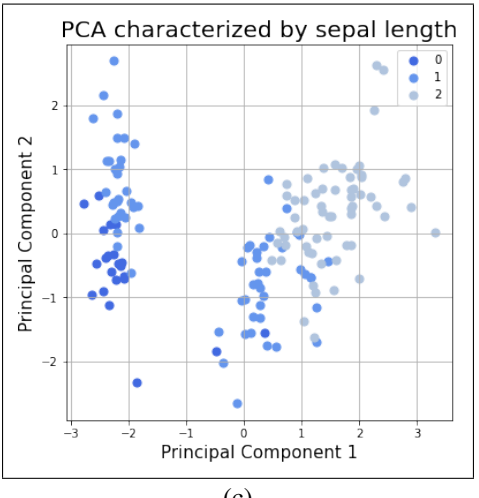
3. **Visualization of Iris dataset through tSNE :**
   t-Distributed Stochastic Neighbor Embedding (t-SNE) is an unsupervised, non-linear technique primarily used for data exploration and visualizing high-dimensional data.The t-SNE algorithm calculates a similarity measure between pairs of instances in the high dimensional space and in the low dimensional space. It then tries to optimize these two similarity measures using a cost function.
   Here two metric for distance calculation have been shown euclidean and minkowski. Refer to Figure 8
   Observation : We can observe here that clusters are easily visible in the 2d plots except in hamming metric(c) wherein the clusters are very close together. In Euclidean



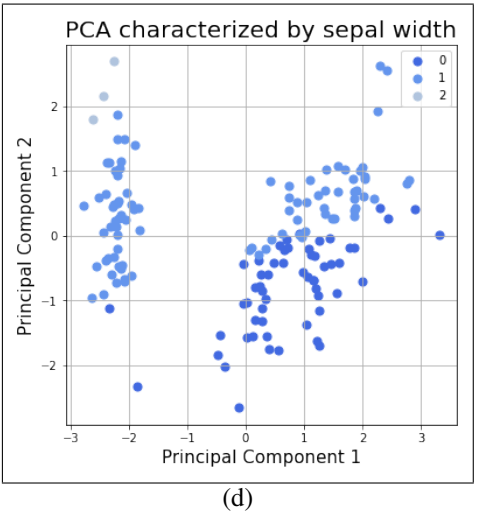
(a)



(b)

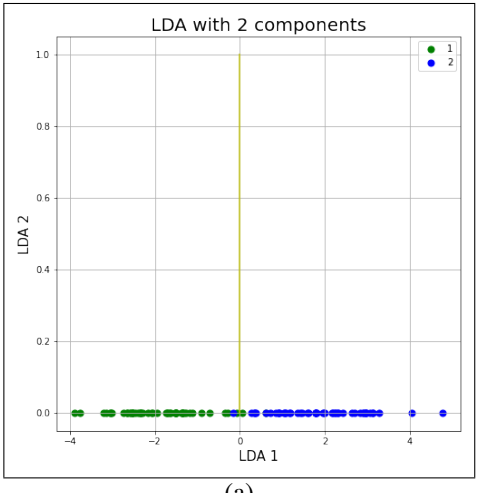


(c)

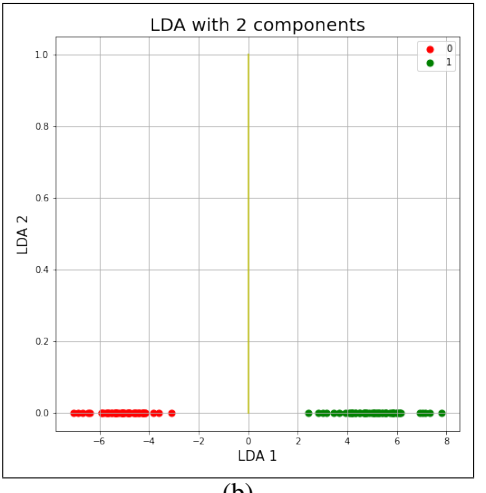


(d)

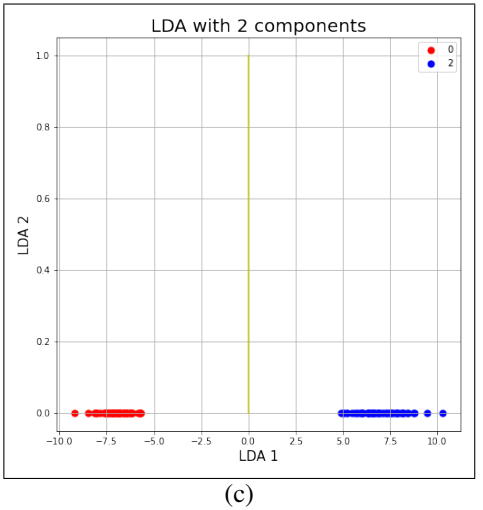
Figure 6: Observing how a single feature is affecting values along an eigen vector.



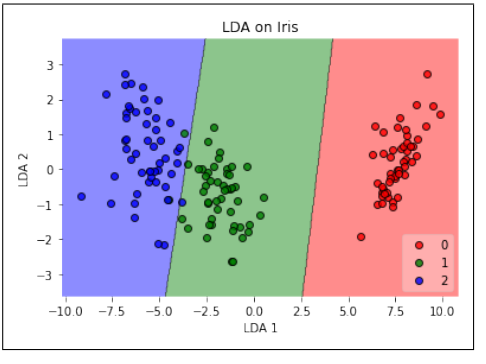
(a)



(b)



(c)



(d)

Figure 7: Visualization through LDA with 2 classes at a time (a,b,c) and taking all 3 classes (d)

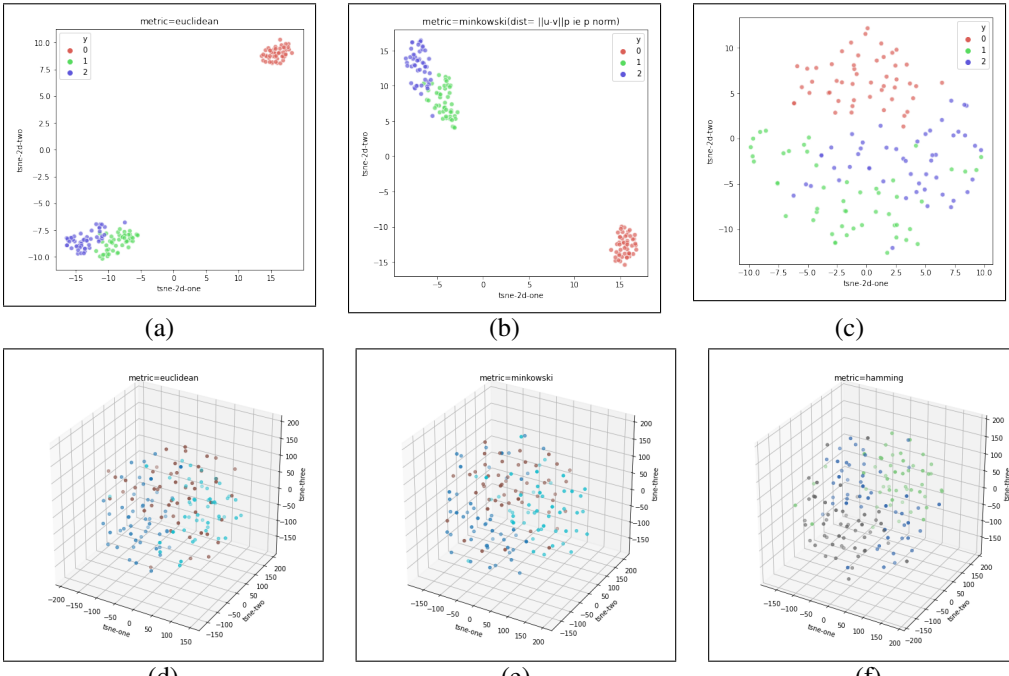


Figure 8: Visualization through LDA with 2 classes at a time (a,b,c) and taking all 3 classes (d)

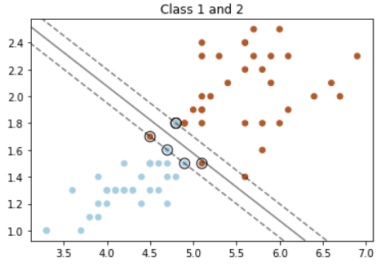
and minkowski metric clusters are same qualitatively just that their position sare different. This is because both metrics measure distance in the same way qualitatively. Although in the 3d plots the clusters are not easily visible from this view but they will become more visible from a side view. Here too some pattern among the three classes is visible to some extent if not fully visible.

1.3 Data Classification with Linear and Non-linear SVMs :

1. **Classification using SVM and visualization of hyperplane** Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection. To plot the line of separation we employ a method where in a margin is added to all possible lines up to the nearest point and then the line of maximum width is selected. The nearest points used here are called **Support Vectors** . In the presented plots in figure 9 circled points represent Support vectors and the dotted lines represent Margins.

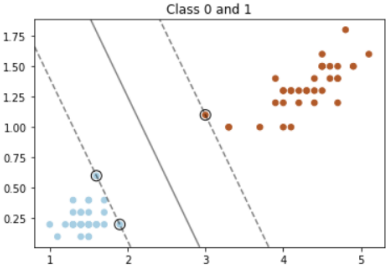
2. **Tuning the C parameter :**
The C parameter is used to harden or soften the margin boundaries. By softening we mean allowing some more points to creep in if that allows a better fit.For very large C, the margin is hard, and points cannot lie in it. For smaller C, the margin is softer, and can grow to encompass some points. Classification report for C=0.001,1,100 are presented in figure 10. We can observe how hardening increases accuracy as it allows

	precision	recall	f1-score	support
1	1.00	0.87	0.93	15
2	0.88	1.00	0.94	15
accuracy			0.93	30
macro avg	0.94	0.93	0.93	30
weighted avg	0.94	0.93	0.93	30



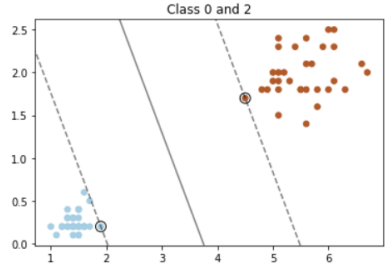
(a)

	precision	recall	f1-score	support
0	1.00	1.00	1.00	16
1	1.00	1.00	1.00	14
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30



(b)

	precision	recall	f1-score	support
0	1.00	1.00	1.00	12
2	1.00	1.00	1.00	18
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30



(c)

Figure 9: Visualization of the max margin hyperplane for every pair of classes alomg with classification report. Dashed line represent margin and circled points represent support vectors.

	precision	recall	f1-score	support
0	0.00	0.00	0.00	19
1	0.41	1.00	0.58	13
2	1.00	1.00	1.00	13
accuracy			0.58	45
macro avg	0.47	0.67	0.53	45
weighted avg	0.41	0.58	0.46	45

	precision	recall	f1-score	support
0	1.00	1.00	1.00	19
1	1.00	1.00	1.00	13
2	1.00	1.00	1.00	13
accuracy			1.00	45
macro avg	1.00	1.00	1.00	45
weighted avg	1.00	1.00	1.00	45

	precision	recall	f1-score	support
0	1.00	1.00	1.00	19
1	1.00	1.00	1.00	13
2	1.00	1.00	1.00	13
accuracy			1.00	45
macro avg	1.00	1.00	1.00	45
weighted avg	1.00	1.00	1.00	45

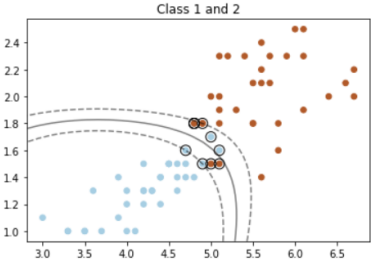
(a)

Figure 10: Classification report for C=0.001(topmost), c=1, c=1000(bottom most)

less opposite class points to creep in. Therefore it is clear that for our dataset we should set a high C value for better results. **However,** we should note here that large C doesn't always guarantee better test results as it may lead to **overfitting**.

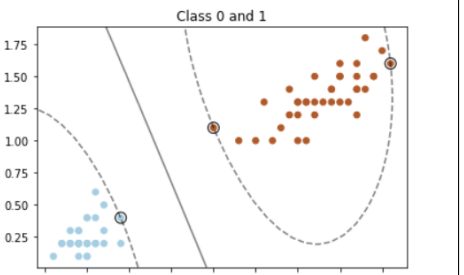
3. **RBF kernel SVM :**
   RBF(Radial Basis Function) allows us to classify data which might not be linearly separable by adding an extra dimension. In the linear model we were getting relatively low accuracy for class 1 & 2 classification because some points among these classes were overlapping as visible from 2D scatter plot. Therefore those points could not be distinguished by just a line.
   Thus by mapping the points to 3D using RBF we are able to classify relatively better and hence accuracy increased 4 percent. However training RBF kernel on larger datasets can become proportionately expensive as the size of dataset increases and hence we should use linear kernel whenever feasible.

	precision	recall	f1-score	support
1	0.93	1.00	0.97	14
2	1.00	0.94	0.97	16
accuracy			0.97	30
macro avg	0.97	0.97	0.97	30
weighted avg	0.97	0.97	0.97	30



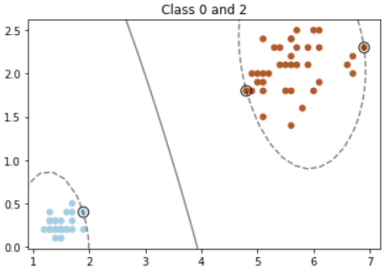
(a)

	precision	recall	f1-score	support
0	1.00	1.00	1.00	18
1	1.00	1.00	1.00	12
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30



(b)

	precision	recall	f1-score	support
0	1.00	1.00	1.00	18
2	1.00	1.00	1.00	12
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30



(c)

Figure 11: Visualization of the max margin hyperplane for every pair of classes along with classification report using **RBF Kernel**. Dashed line represent margin and circled points represent support vectors.

1.4 Conclusion

We saw how dimensionality reduction is a very important tool for visualization of data but in some cases it may not give expected results or actuak underlying patterns as we saw in the face recognition problem when no visible clusters were formed. Then we saw how Support vectors are a great tool to select the optimum line of separation. We observed that reducing no of components during PCA can affect accuracy to a great extent as many unique distinguishin features can be lost in the process.

1.5 References

1. https://en.wikipedia.org/wiki/Kernel_method
2. https://en.wikipedia.org/wiki/Eigenface
3. https://towardsdatascience.com/svm-and-kernel-svm-fed02bef1200
4. https://towardsdatascience.com/visualising-high-dimensional-datasets-using-pca-and-t-sne-in-python-8ef87e7915b
5. https://scikit-learn.org/stable/auto_examples/svm/plot_separating_hyperplane.html
6. https://towardsdatascience.com/an-introduction-to-t-sne-with-python-example-5a3a29310
7. https://towardsdatascience.com/face-recognition-using-deep-learning-b9be73689a23
8. https://scikit-learn.org/stable/auto_examples/applications/plot_face_recognition.html