Dimensionality Reduction & SVMs

Shubham Pradhan
2017EEB1168

Indian Institute of Technology
Ropar, Punjab
India

1 Introduction

In machine learning we are having too many factors on which the final classification is done. These factors are basically, known as variables. The higher the number of features, the harder it gets to visualize the training set and then work on it. Sometimes, most of these features are correlated, and hence redundant. This is where dimensionality reduction algorithms come into play. When we deal with real problems and real data we often deal with high dimensional data that can go up to millions. In original high dimensional structure, data represents itself. Although, sometimes we need to reduce its dimensionality. We need to reduce the dimensionality that needs to associate with visualizations. Although, that is not always the case.

2 Techniques of dimensionality reduction

2.1 Principal Component Analysis

The main linear technique for dimensionality reduction, principal component analysis, performs a linear mapping of the data to a lower-dimensional space in such a way that the variance of the data in the low-dimensional representation is maximized. In practice, the covariance (and sometimes the correlation) matrix of the data is constructed and the eigenvectors on this matrix are computed. The eigenvectors that correspond to the largest eigenvalues (the principal components) can now be used to reconstruct a large fraction of the variance of the original data. Moreover, the first few eigenvectors can often be interpreted in terms of the large-scale physical behavior of the system, because they often contribute the vast majority of the system's energy, especially in low-dimensional systems.

2.2 Linear discriminant analysis

LDA is also closely related to principal component analysis (PCA) in that they both look for linear combinations of variables which best explain the data. LDA explicitly attempts to model the difference between the classes of data. PCA, in contrast, does not take into account any difference in class, and factor analysis builds the feature combinations based on differences rather than similarities. Discriminant analysis is also different from factor analysis in that it is not an interdependence technique: a distinction between independent variables and dependent variables (also called criterion variables) must be made.

2.3 T-Distributed Stochastic Neighbor Embedding

T-Distributed Stochastic Neighbor Embedding (t-SNE) is an unsupervised, non-linear technique primarily used for data exploration and visualizing high-dimensional data. In simpler terms, t-SNE gives you a feel or intuition of how the data is arranged in a high-dimensional space. 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.

3 Task 1: Principal Component Analysis and Eigenfaces for Face Recognition

Eigenfaces was proposed by Turk and Pentland in the 1980s for face recognition, and represent one of the early breakthroughs in the field of Computer Vision. The trick in this approach involves significantly reducing the dimensionality of a high-dimensional vector (a 320 x 240 pixel image can be viewed as a 76800-dimensional vector) to far fewer dimensions employing Principal Component Analysis, and being able to reconstruct each training image as a linear combination of eigen-faces, which resemble ghost faces.

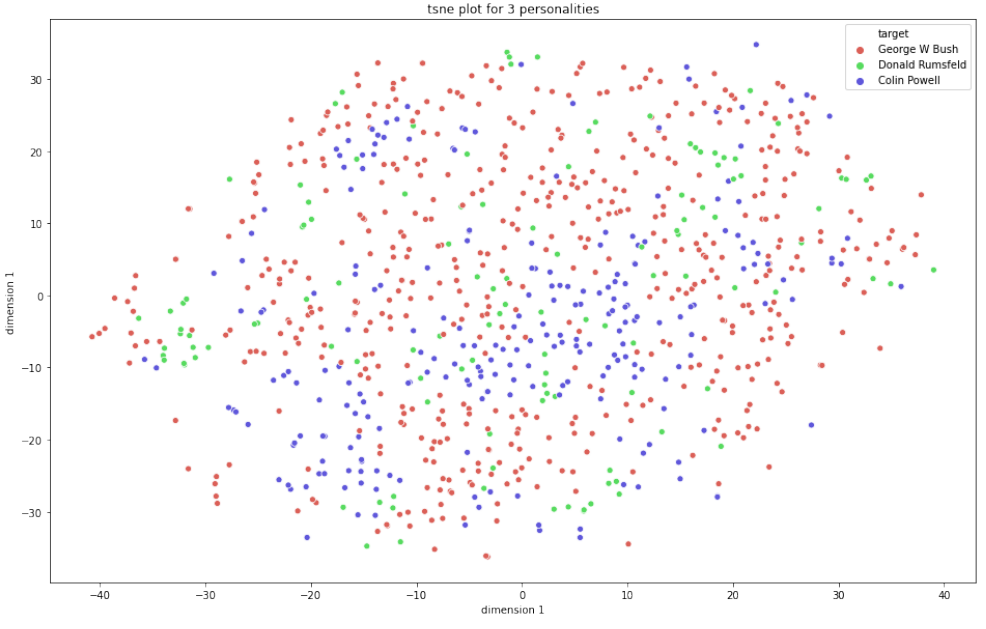
3.1 Part 1

We first did the test train split if the dataset with test size = 0.3. Then successfully implemented principal component analysis on "Labeled Faces in the Wild" dataset on both training and testting set individually. We reduced the dimensions from 2914 to 100. We took top 100 principal component corresponding to the top 100 Eigen values. Variance along each of the 100 dimension is hown below.

```
Variance explained by each Dimension is
[770883.9    643581.06   287779.75   246428.06   224148.08   128750.13
 103473.      89088.31    86318.13    75501.75    66431.66    57432.44
  50215.895   44220.816   41938.812   40215.043   37304.637   35875.605
  32350.354   30167.021   28027.844   26014.95    25349.707   24180.61
  23792.143   21117.248   20621.6     19610.777   18956.172   18024.258
  16747.797   16246.775   15360.206   14348.866   14137.735   13781.645
  13504.026   12691.488   12395.42    12056.26    11844.35    11583.702
  10866.006   10500.355   10340.977   10210.034   10106.35     9841.174
   9556.962    9139.924    8954.025    8467.261    8167.018    8073.374
   7965.07     7746.328    7516.6445   7406.872    7211.009    7118.6064
   7003.886    6726.0156   6661.201    6581.121    6482.8804   6421.2627
   6154.091    6040.7715   6005.026    5867.5244   5658.8696   5558.0615
   5416.726    5223.088    5187.485    5046.0356   5001.2217   4926.9424
   4911.376    4844.414    4707.102    4641.4106   4624.976    4502.872
   4460.797    4342.451    4267.7725   4172.126    4085.809    4042.6863
   4021.1829   3935.8914   3854.2056   3696.3083   3660.9924   3588.0408
   3555.6675   3445.7656   3415.065    3345.0117]
```

3.2 Part 2

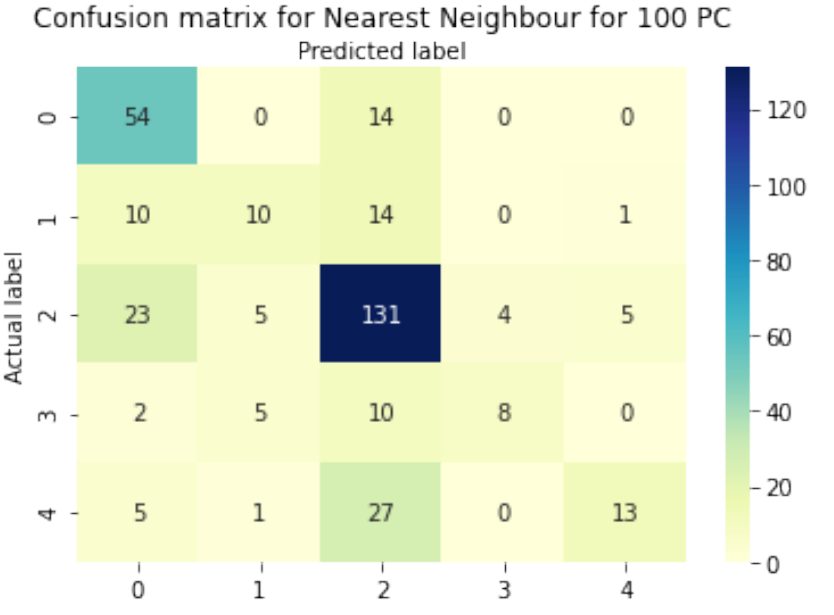
I project each face in the training and test datasets onto the 100-dimensional eigenface space. This transformed each face image to a 100-D vector. Then i Chose three personalities from the train set and plotted the points corresponding to the train+test faces for these identities after projecting them to 2 dimensions via tSNE. The plot is shown below.



It is observed from the scatter plot that no clustering of the classes took place. All the datapoints are randomly distributed across the plot. Reason behind this may be due the presence of lots of outliers or not clustering in original dataset in original dimensions.

3.3 Part 3

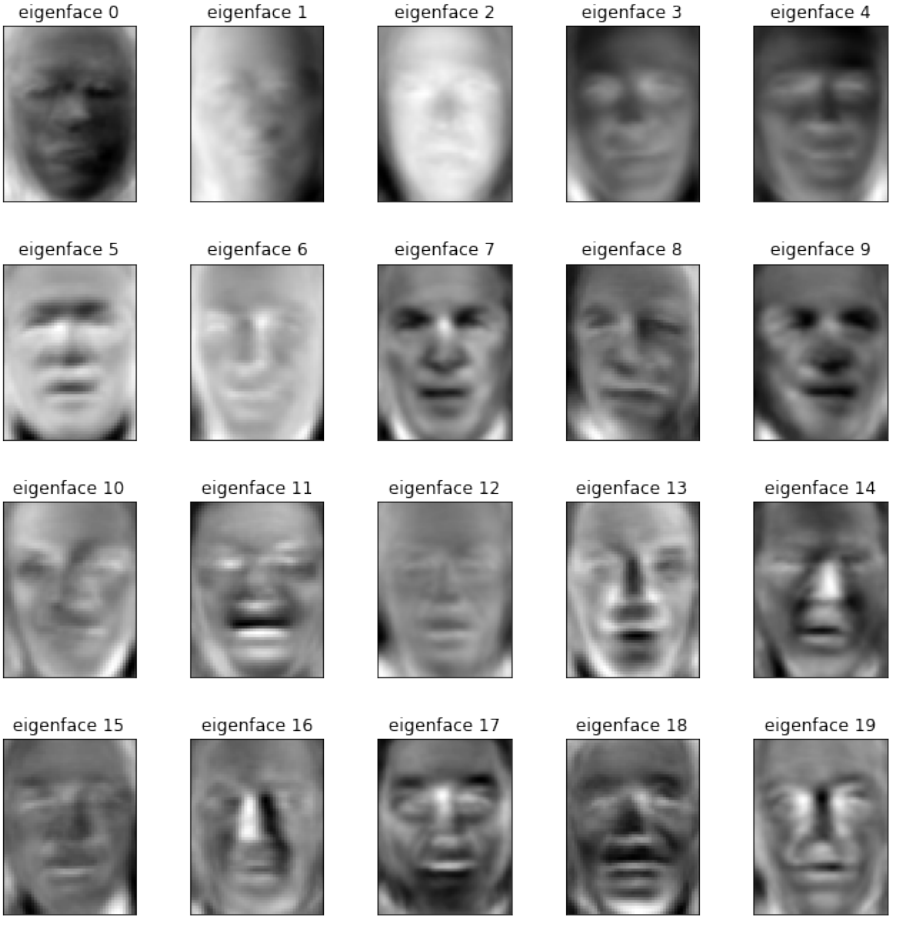
We implemented the Nearest Neighbour classifier algorithm on the dataset. We implemented it on the dataset obtained in the first part i.e. on 100 dimension got from PCA. Coming to Nearest Neighbour Classifier, it is clear from the term itself that for every test image we compare it with every labeled trained data to predict its label. Here we used 5 nearest neighbour to classify each point. Similar to nearest neighbour classifier, it does all the same work but among its k nearest neighbours. The label occuring with most frequency is the label for the test image. Classification Report is shown below.



	precision	recall	f1-score	support
Colin Powell	0.57	0.79	0.67	68
Donald Rumsfeld	0.48	0.29	0.36	35
George W Bush	0.67	0.78	0.72	168
Gerhard Schroeder	0.67	0.32	0.43	25
Tony Blair	0.68	0.28	0.40	46
accuracy			0.63	342
macro avg	0.61	0.49	0.52	342
weighted avg	0.63	0.63	0.61	342

3.4 Part 4

In this part we plotted the Eigen Faces of the given dataset. These eigen faces are shown below.

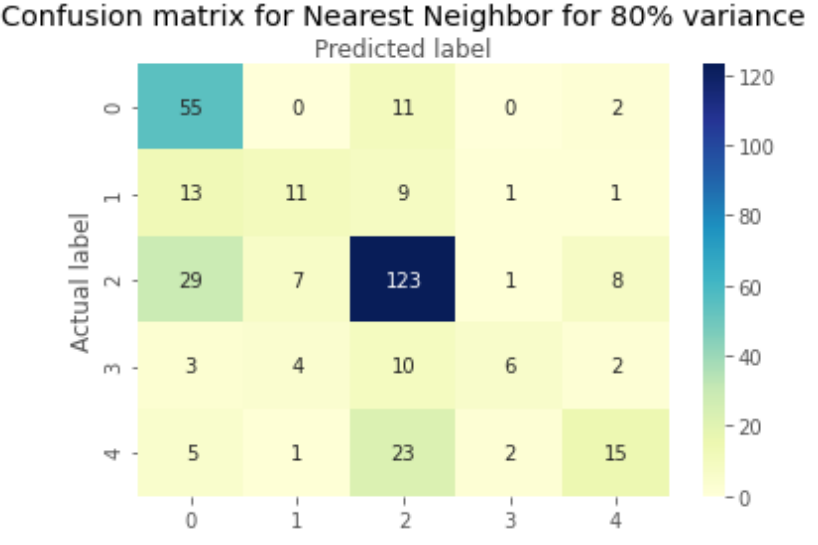


One interesting part of PCA is that it computes the "mean" face. The principal components measure deviations about this mean along orthogonal axes. The components ("eigenfaces") are ordered by their importance from top-left to bottom-right. We see that the first few components seem to primarily take care of lighting conditions; the remaining components pull out certain identifying features: the nose, eyes, eyebrows, etc.

3.5 Part 5

Instead of hardcoding the number of PCA components, we could also retain as many eigenvectors so that a certain amount of the original data variance is preserved. In this example we determined how many eigenfaces (let us say this number is x) are required to retain 80% variance of the original training set. X turned out to be 30. Then we projected original

dataset on the 30 dimension and repeated the nearest neighbour classifier on these dimensions. Classification report is shown below.



	precision	recall	f1-score	support
Colin Powell	0.52	0.81	0.64	68
Donald Rumsfeld	0.48	0.31	0.38	35
George W Bush	0.70	0.73	0.72	168
Gerhard Schroeder	0.60	0.24	0.34	25
Tony Blair	0.54	0.33	0.41	46
accuracy			0.61	342
macro avg	0.57	0.48	0.50	342
weighted avg	0.61	0.61	0.60	342

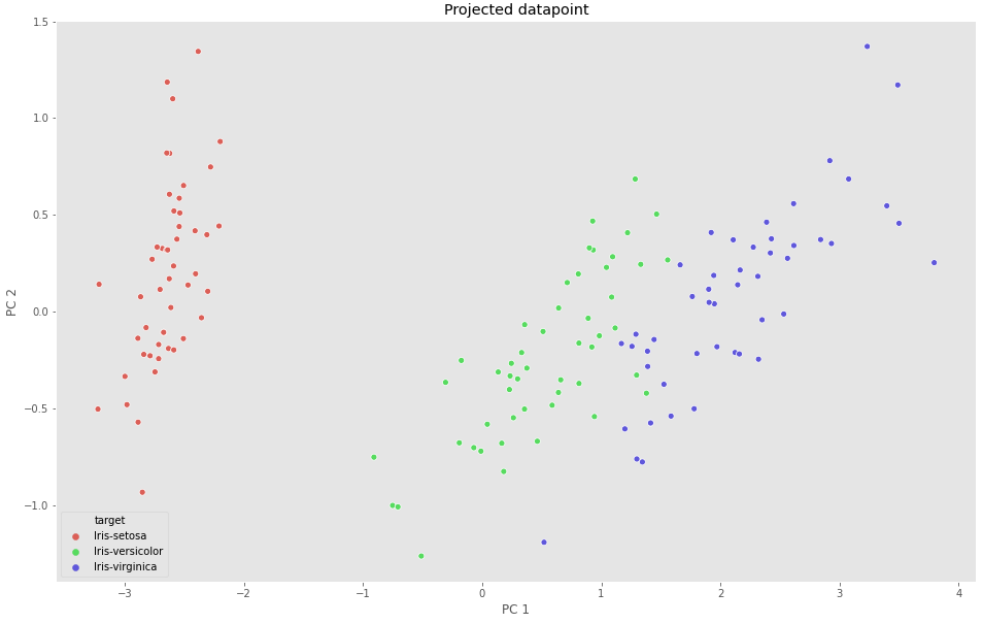
We can compare the results with the results obtained in part 3. We can observe that the Classifier performs better in part 3. It gives better accuracy in part 3. This is because we took more number of eigen values in part 3, this helps in retaining more variance in the features thus results in better classification.

4 Tast 2: Dimensionality Reduction and Visualization with PCA, LDA and tSNE

We will attempt to employ the various dimensionality reduction techniques to visualize the Fisher Iris dataset in this section. Fisher Iris comprises 150 4-dimensional data samples arising from three Iris flower species. The four features measured for each data sample are: sepal length and width, petal length and width.

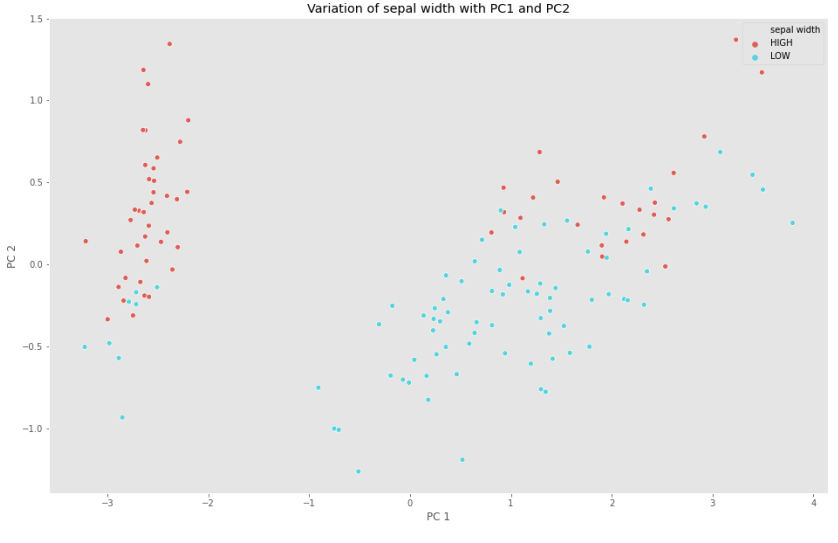
4.1 Part 1

Here I projected the 4 dimensional data ti 2 dimensions using PCA. The Variance along each of the principal component is [4.22484077 0.24224357]. The projected datapoints are shown below.

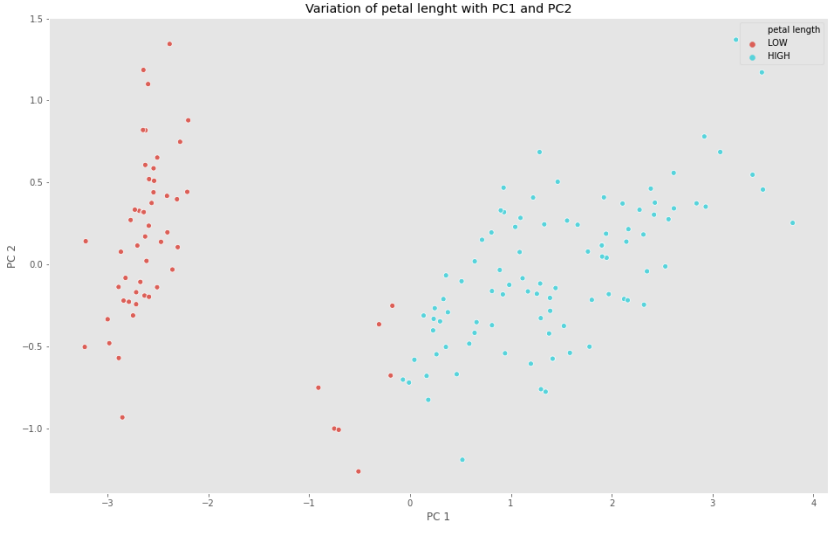


Then to explain the data variation along 1st and 2nd principal component I divided the features in two classes that are high values and low values. I calculated the mean of each of the 4 feature and then classified the values grater than mean as HIGH and values lower than mean as LOW. Variation along each feature is shown below.

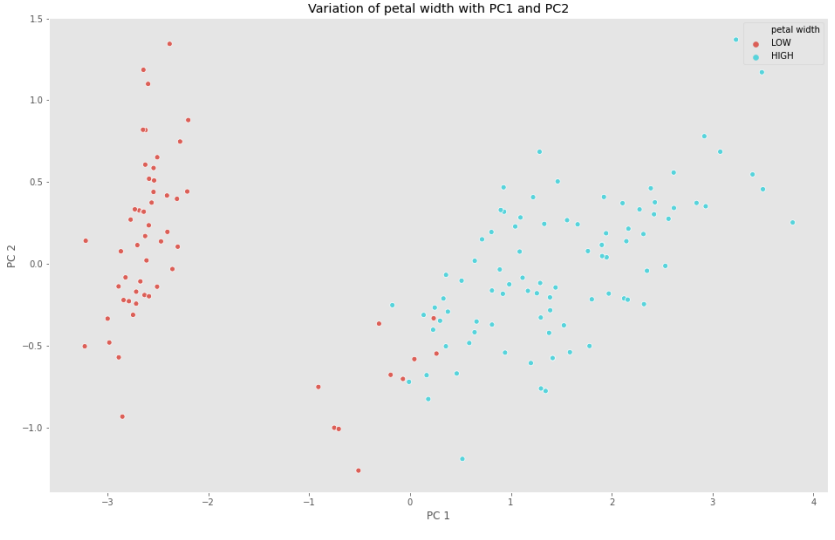
From above graph it can be seen that with high values of Sepal length correspond to high values of PC1 and PC2.



From above graph it can be observed that with high values of Sepal width correspond to high value of PC2 but variation of sepal width along PC1 does give any pattern.



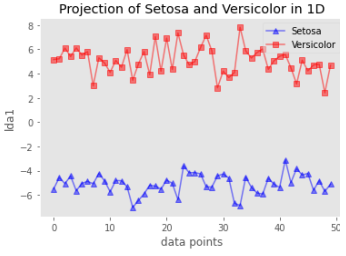
From above graph it can be observed that with high values of Petal Length correspond to high value of PC1 but variation of petal length along PC2 does give any pattern.

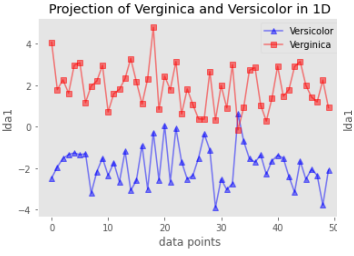


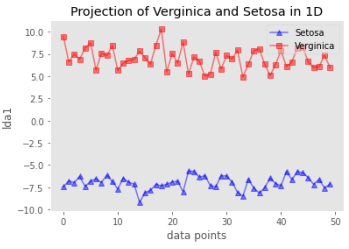
From above graph it can be observed that with high values of Petal width correspond to high value of PC1 but variation of petal width along PC2 does give any pattern.

4.2 Part 2

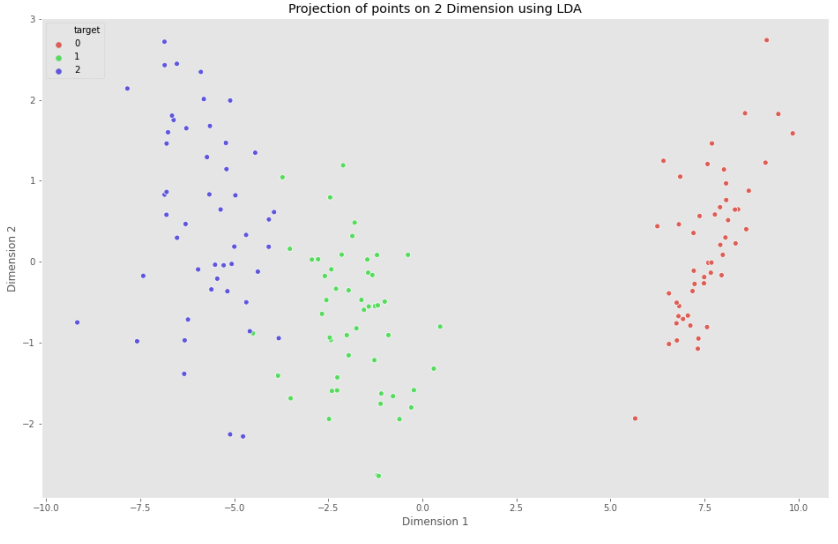
LDA adopts a supervised approach to dimensionality-reduction by projecting labelled C class data onto a C-1 dimensional space. We considered two classes at a time and projected it to 1 dimension. For each of the three pair the projections are shown below The line if seperation between two classes is given by lda1=0.







Then we pro projected 3-class Fisher Iris data on the 2D feature space that maximizes inter-class scatter while minimizing intra-class scatter.



4.3 Part 3

In this we projected 4 dimensional data upon 2 dimension and 3 dimension seperatly. For both transformations we varied 2 metric parameter to get different plots. These parameter were perplexity and learning rate. The perplexity is related to the number of nearest neighbors that is used in other manifold learning algorithms. Larger datasets usually require a larger perplexity. he learning rate for t-SNE is usually in the range [10.0, 1000.0]. If the learning rate is too high, the data may look like a 'ball' with any point approximately equidistant from its nearest neighbours. If the learning rate is too low, most points may look compressed in a dense cloud with few outliers.

Below are the graphs for variation of perplexity for both 2d transformations.

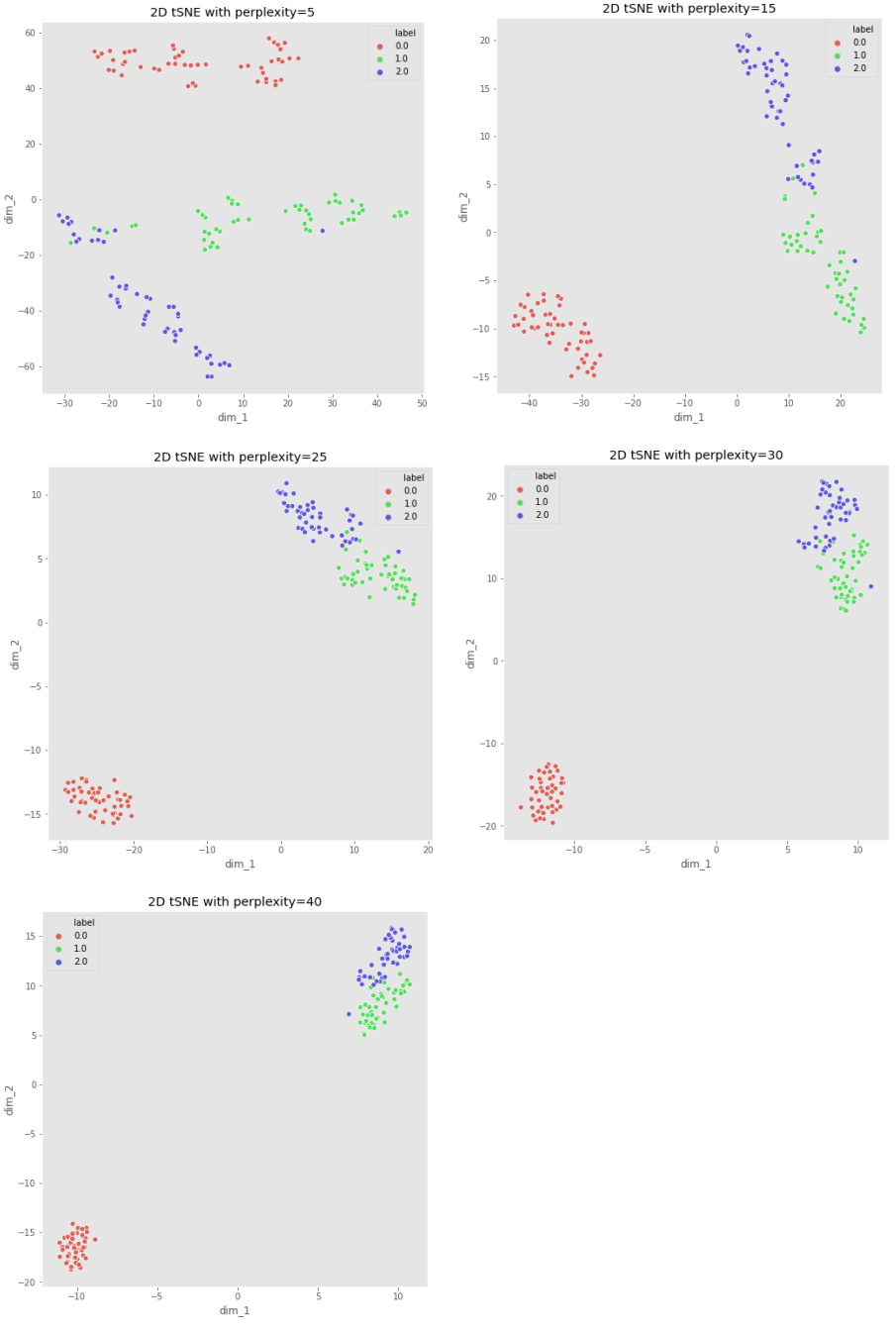


Figure 1: Variation of perplexity in 2D transformation
As the perplexity increases cluster becomes smaller and compact.

Below are the graphs for variation of perplexity for both 3d transformations.

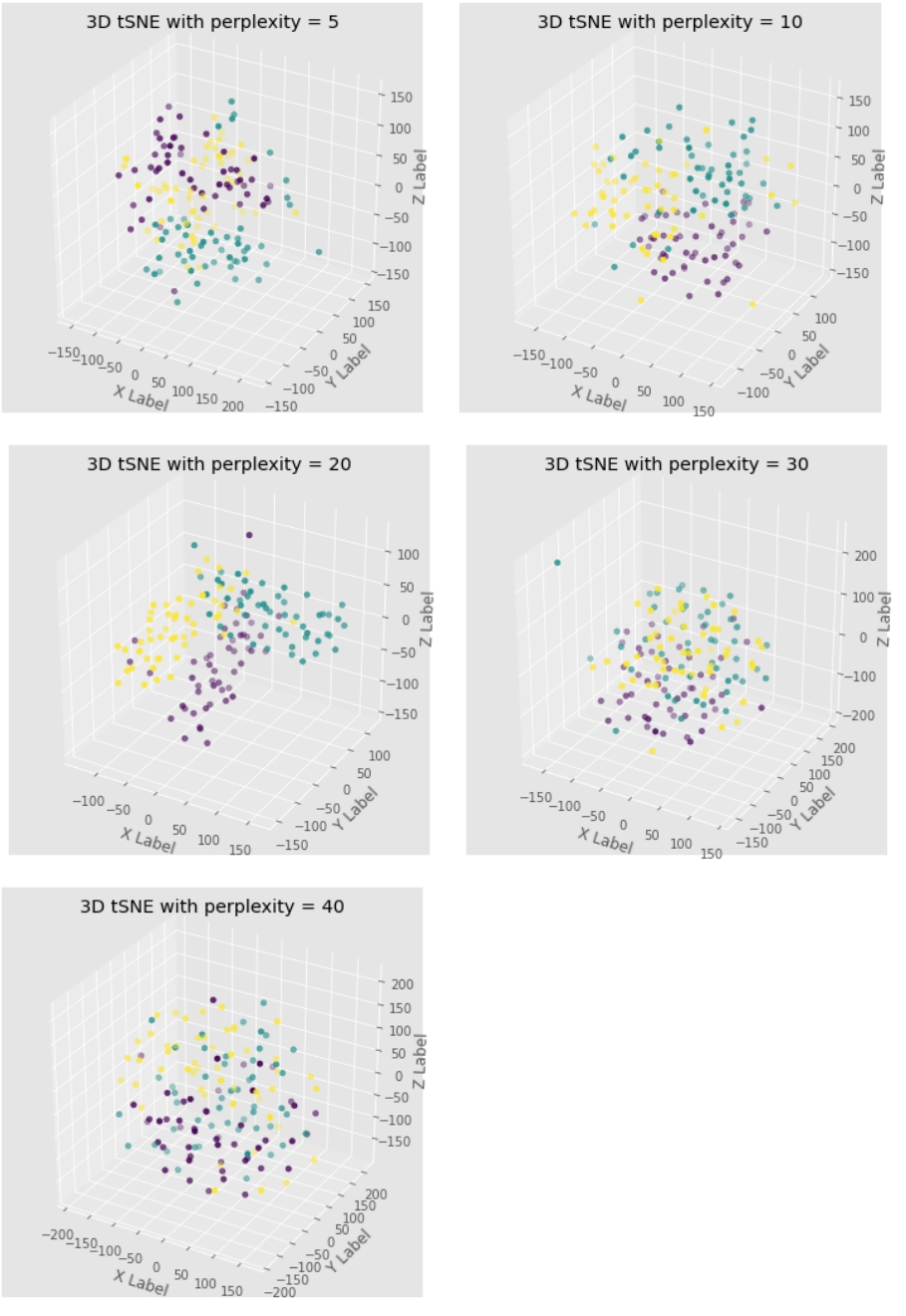
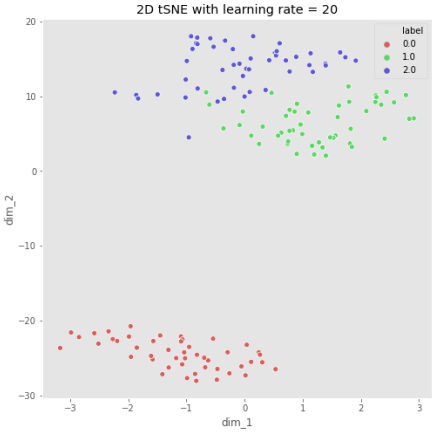
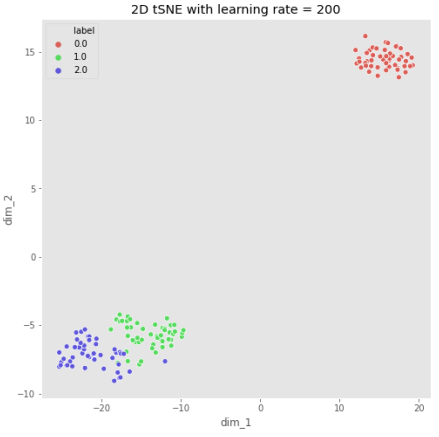
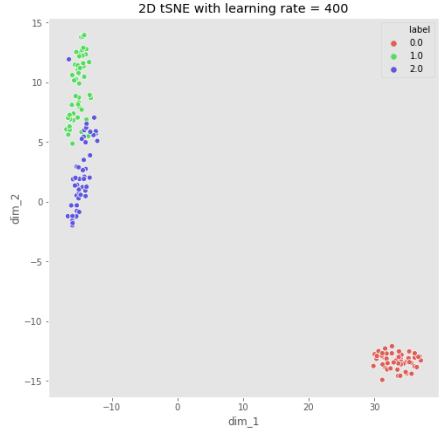


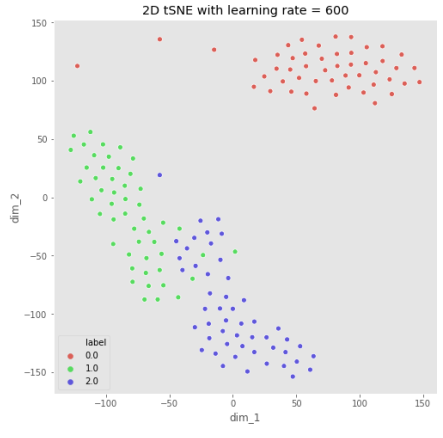
Figure 2: Variation of perplexity in D transformation

Below are the graphs for variation of learning rate for 2d transformations.









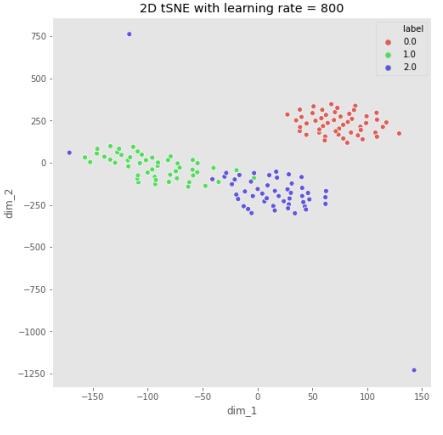


Figure 3: Variation of learning rate in 2D transformation

As the the learning rate increases the clusters become compact but after a certain value the cluster again start expanding.

Below are the graphs for variation of perplexity for 3d transformations.

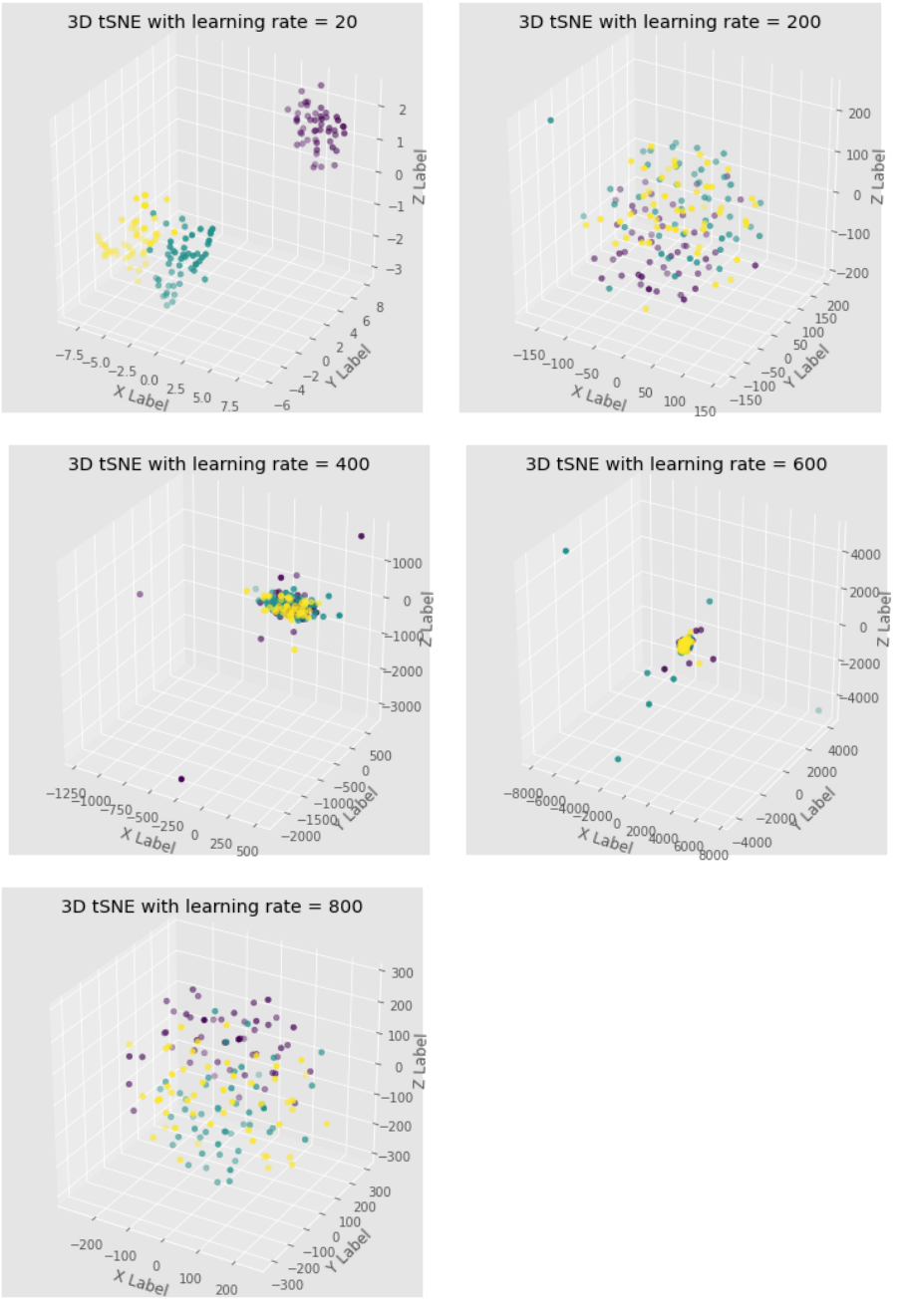


Figure 4: Variation of learning rate in 3D transformation

5 Tast 3: Data Classification with Linear and Non-linear SVMs

Support Vector Machine, abbreviated as SVM can be used for both regression and classification tasks. But, it is widely used in classification objectives. The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space(N — the number of features) that distinctly classifies the data points.

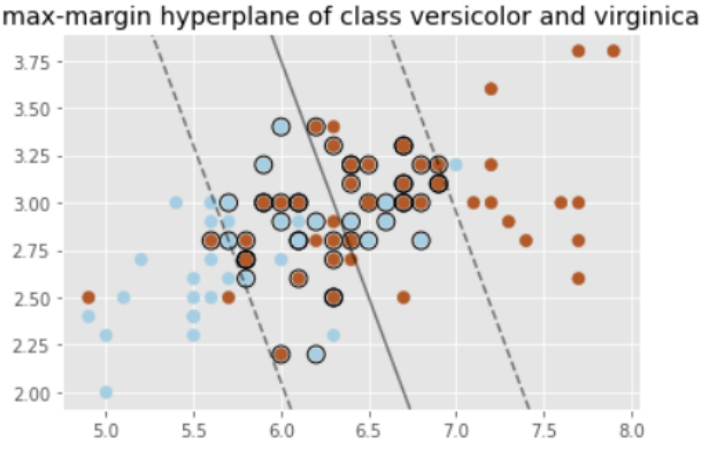
5.1 Part 1

In this part we implemented the SVM algorithm with linear kernel on the dataset. We took two classes at a time and predicted the classes. Classification report for all the 3 pair of classes are shown below. In this part we took the value of C=1.

classification report of setosa and virginica

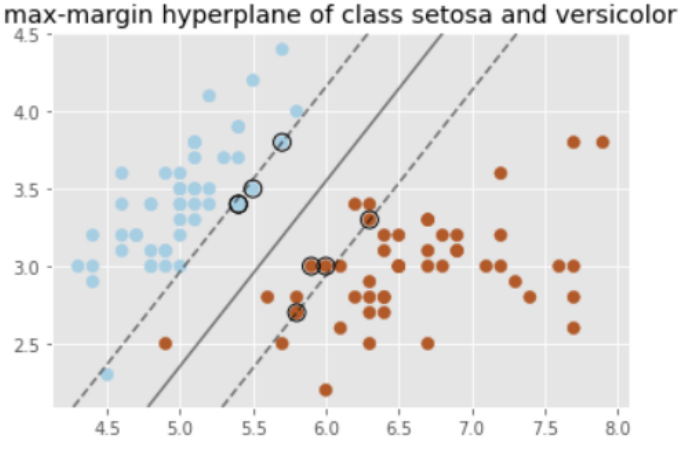
	precision	recall	f1-score	support
Iris setosa	1.00	1.00	1.00	18
Iris virginica	1.00	1.00	1.00	12
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30

Thus we can see that linear SVM was very well able to classify between setosa and versicolor and between setosa and virginica but could not perform well for versicolor and virginica.



classification report of versicolor and virginica

	precision	recall	f1-score	support
Iris virginica	0.72	0.93	0.81	14
Iris versicolor	0.92	0.69	0.79	16
accuracy			0.80	30
macro avg	0.82	0.81	0.80	30
weighted avg	0.83	0.80	0.80	30



classification report of setosa and versicolor

	precision	recall	f1-score	support
Iris setosa	0.92	1.00	0.96	11
Iris versicolor	1.00	0.95	0.97	19
accuracy			0.97	30
macro avg	0.96	0.97	0.96	30
weighted avg	0.97	0.97	0.97	30

5.2 Part 2

The standard SVM classifier function assumes a C value of 1. In this we part we implemented SVM classifier with C value 0.001 and 1000 for each pair of class. All classification report are shown below.

classification report of setosa and virginica for c=0.001

	precision	recall	f1-score	support
Iris setosa	0.47	1.00	0.64	14
Iris virginica	0.00	0.00	0.00	16
accuracy			0.47	30
macro avg	0.23	0.50	0.32	30
weighted avg	0.22	0.47	0.30	30

classification report of setosa and virginica for c = 1000

	precision	recall	f1-score	support
Iris setosa	1.00	1.00	1.00	14
Iris virginica	1.00	1.00	1.00	16
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30

classification report of versicolor and virginica for c =0.001

	precision	recall	f1-score	support
Iris virginica	0.00	0.00	0.00	17
Iris versicolor	0.43	1.00	0.60	13
accuracy			0.43	30
macro avg	0.22	0.50	0.30	30
weighted avg	0.19	0.43	0.26	30

classification report of versicolor and virginica for c= 1000

	precision	recall	f1-score	support
Iris virginica	0.92	0.71	0.80	17
Iris versicolor	0.71	0.92	0.80	13
accuracy			0.80	30
macro avg	0.81	0.81	0.80	30
weighted avg	0.83	0.80	0.80	30

Here we can see that the higher the value of C the much better is the classification. This is because Large Value of parameter C imply small margin and Small Value of parameter C imply Large margin.

The C parameter tells the SVM optimization how much you want to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-

margin separating hyperplane, even if that hyperplane misclassifies more points. For very tiny values of C, you should get misclassified examples, often even if your training data is linearly separable.

classification report of setosa and versicolor for c =0.001

	precision	recall	f1-score	support
Iris setosa	0.83	1.00	0.91	15
Iris versicolor	1.00	0.80	0.89	15
accuracy			0.90	30
macro avg	0.92	0.90	0.90	30
weighted avg	0.92	0.90	0.90	30

classification report of setosa and versicolor for c =1000

	precision	recall	f1-score	support
Iris setosa	1.00	1.00	1.00	15
Iris versicolor	1.00	1.00	1.00	15
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30

5.3 Part 3

The RBF kernel SVM decision region is actually also a linear decision region. What RBF kernel SVM actually does is to create non-linear combinations of your features to uplift your samples onto a higher-dimensional feature space where you can use a linear decision boundary to separate your classes. The gamma parameter defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’. The gamma parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors.

In this part we implement SVM with RBF kernel instead of linear kernel on each pair of classes. The classification report for each pair of classes with varying C parameter is shown below along with hyperplane representation.

classification report of setosa and virginica with kernel = rbf

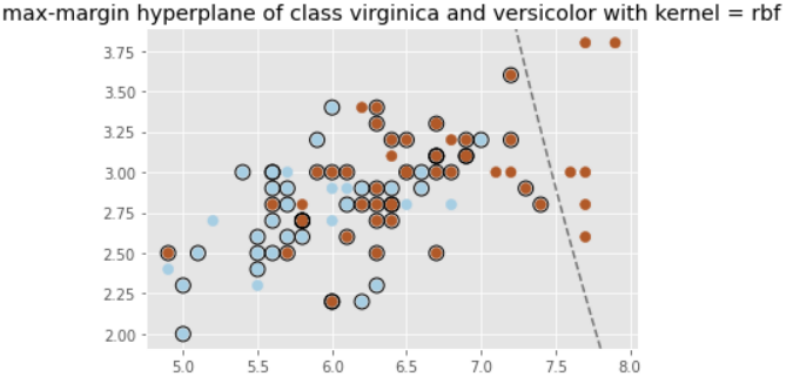
	precision	recall	f1-score	support
Iris setosa	0.74	1.00	0.85	14
Iris virginica	1.00	0.69	0.81	16
accuracy			0.83	30
macro avg	0.87	0.84	0.83	30
weighted avg	0.88	0.83	0.83	30

classification report of setosa and virginica with kernel = rbf & c = 1000

	precision	recall	f1-score	support
Iris setosa	1.00	1.00	1.00	14
Iris virginica	1.00	1.00	1.00	16
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30

classification report of setosa and virginica with kernel = rbf & c =0.001

	precision	recall	f1-score	support
Iris setosa	0.47	1.00	0.64	14
Iris virginica	0.00	0.00	0.00	16
accuracy			0.47	30
macro avg	0.23	0.50	0.32	30
weighted avg	0.22	0.47	0.30	30



classification report of virginica and versicolor with kernel = rbf

	precision	recall	f1-score	support
Iris virginica	0.00	0.00	0.00	17
Iris versicolor	0.43	1.00	0.60	13
accuracy			0.43	30
macro avg	0.22	0.50	0.30	30
weighted avg	0.19	0.43	0.26	30

classification report of virginica and versicolor with kernel = rbf & c = 1000

	precision	recall	f1-score	support
Iris virginica	0.92	0.71	0.80	17
Iris versicolor	0.71	0.92	0.80	13
accuracy			0.80	30
macro avg	0.81	0.81	0.80	30
weighted avg	0.83	0.80	0.80	30

classification report of virginica and versicolor with kernel = rbf & c =0.001

	precision	recall	f1-score	support
Iris virginica	0.00	0.00	0.00	17
Iris versicolor	0.43	1.00	0.60	13
accuracy			0.43	30
macro avg	0.22	0.50	0.30	30
weighted avg	0.19	0.43	0.26	30

classification report of setosa and versicolor with kernel = rbf

	precision	recall	f1-score	support
Iris setosa	1.00	1.00	1.00	15
Iris versicolor	1.00	1.00	1.00	15
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30

classification report of setosa and versicolor with kernel = rbf & c = 1000

	precision	recall	f1-score	support
Iris setosa	1.00	0.93	0.97	15
Iris versicolor	0.94	1.00	0.97	15
accuracy			0.97	30
macro avg	0.97	0.97	0.97	30
weighted avg	0.97	0.97	0.97	30

classification report of setosa and versicolor with kernel = rbf & c =0.001

	precision	recall	f1-score	support
Iris setosa	0.88	1.00	0.94	15
Iris versicolor	1.00	0.87	0.93	15
accuracy			0.93	30
macro avg	0.94	0.93	0.93	30
weighted avg	0.94	0.93	0.93	30

Here also we can see that as the value of C increases there is better classification. The C parameter trades off correct classification of training examples against maximization of the decision function's margin. For larger values of C, a smaller margin will be accepted if the decision function is better at classifying all training points correctly. A lower C will encourage a larger margin, therefore a simpler decision function, at the cost of training accuracy. In other words"C" behaves as a regularization parameter in the SVM.

6 Conclusion

In this assignment we implemented three different techniques for dimensionality reduction. We implemented PCA, LDA and t-SNE. t-SNE is majorly used only for the data visualisation. PCA is the unsupervised linear algorithm. Its one of the application is in finding Eigen faces for face detection. LDA is also linear algorithm but it is supervised algorithm. We also implemented K-Nearest Neighbour classifier which is the supervised learning algorithm. At the end we implemented SVM which is also supervised learning algorithm and varied its different parameter for different performance. Thus all tha algorithms were successfully implemented in this assignment.