Assignment 3

Preetesh Verma
2018EEB1171

Abstract

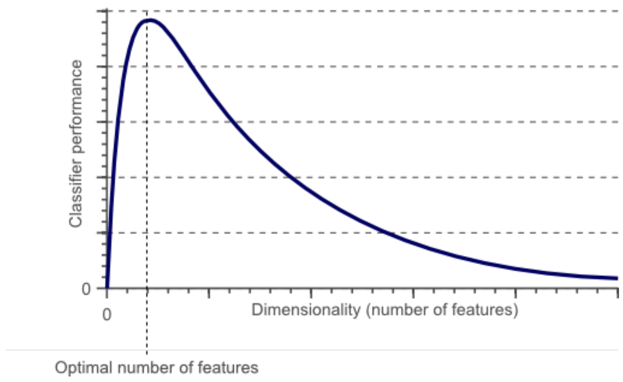
This document demonstrates the various observations made from the third lab assignment given in the course CS521 i.e. Fundamentals of Data Science.The assignment was divided into three parts with the first part focusing on the implementation of eigen faces concept used in the field of face recognition in the field of Computer Vision using PCA and KNN i.e. k Nearest Neighbour algorithm to classify the test faces using the results from the training set. The dataset used for this task is “Labeled Faces in the Wild”. The second part of the assignment was to use various dimension reduction techniques(such as TSNE,LDA,PCA) on the famous iris dataset and to draw conclusions from the observations and perform classification.The last part focuses on the usage of the Support Vector Machine algorithm to perform classification on the iris dataset but only considering two dimensions i.e. either sepal dimensions or the petal dimensions.

At each level of the assignment proper observations were to be made from the result and conclusions were to be drawn thereafter.The task also involved changing the values of various hyper parameters and check the differences that occur because of that.

Therefore, the main tasks and learning’s of the assignment focuses on dimensionality reduction and classification.

1 About Dimensionality Reduction

Often in Machine Learning problems we are having more than enough number of dimensions which can cause our algorithms to overfit the training data because the model would be much more complex when the number of the features is significantly high and as such increase our variance, so we cannot get satisfactory results **THE CURSE OF DIMENSIONALITY.**.As the number of features increase, the number of samples also increases proportionally.



So to tackle such problems we generally do the dimensionality reduction of our dataset i.e. reduce the number of features in our dataset and also trying to retain as much essential information as possible from the dataset.In statistics, machine learning, and information theory, dimensionality reduction or dimension reduction is the process of reducing the number of random variables under consideration by obtaining a set of principal variables. Thus dimension reduction also helps is in reducing the computational costs in performing the various tasks on the dataset as the number of features are now considerably less so space and time required also reduces.There are two main approaches to achieve dimensionality reduction which are **feature selection and feature extraction.**
Feature selection: In this, we try to find a subset of the original set of variables, or features, to get a smaller subset which can be used to model the problem. It usually involves three ways:
Filter,Wrapper and Embedded
Feature extraction: This reduces the data in a high dimensional space to a lower dimension space, i.e. a space with lesser no. of dimensions.These involve PCA,TSNE and LDA.

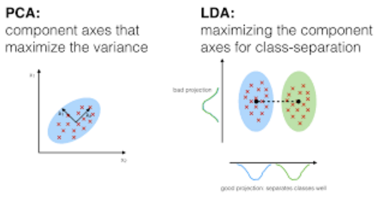
1.1 PCA

Principal Components Analysis, or PCA, is a unsupervised data analysis tool that is usually used to reduce the dimensionality (number of variables) of a large number of interrelated variables, while retaining as much of the information (variation) as possible. PCA calculates an uncorrelated set of variables (components or pc's). These components are ordered so that the first few retain most of the variation present in all of the original variables. Unlike its cousin Factor Analysis, PCA always yields the same solution from the same data (apart from arbitrary differences in the sign) i.e. it is deterministic. The computations of PCA reduce to an eigenvalue-eigenvector problem. $\mathbf{Sa} - \lambda\mathbf{a} = \mathbf{0} \Longleftrightarrow \mathbf{Sa} = \lambda\mathbf{a}.$

PCA always tries to maximize the variance which serves as the condition while the magnitude being equal to 1 as the constraint in the Lagrange Multiplier optimization technique. Eigenvectors denote the directions of greatest variance and eigen values the importance of direction. Larger the variance more important is the direction.

1.2 LDA

Linear Discriminant Analysis (LDA) is a supervised dimensionality reduction technique. As the name implies dimensionality reduction techniques reduce the number of dimensions (i.e. variables) in a dataset while retaining as much information as possible. Compute the within class and between class scatter matrices and Compute the eigenvectors and corresponding eigenvalues for the scatter matrices.Sort the eigenvalues and select the top k and then create a new matrix containing eigenvectors that map to the k eigenvalues. Obtain the new features (i.e. LDA components) by taking the dot product of the data and the matrix of k eigen vectors.


PCA:
component axes that maximize the variance
LDA:
maximizing the component axes for class-separation
bad projection
good projection: separates classes well

1.3 t-SNE

An unsupervised, non-linear dimension reduction technique.It works pretty well on manifold data as well.

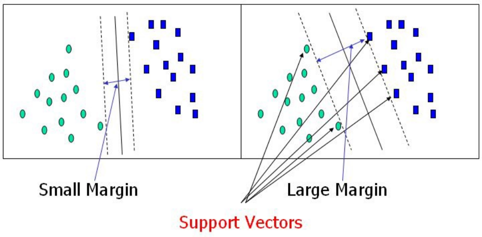
t-SNE tries to preserve small pairwise distances while PCA is concerned with preserving large pairwise distance to maximize variance.The t-SNE algorithm calculates the similarity measures between pairs of instances in high dimensional and low dimensional space and tries to optimize these similarities using Kullback-Liebler divergence.The final result produced by t-SNE is random or stochastic while producing superior visualization as compared to PCA.

2 Classification Problem

In machine learning and statistics, classification is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. Examples are assigning a given email to the "spam" or "non-spam" class. Classification is a pattern recognition problem. Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y). The output variables are often called labels or categories. The mapping function predicts the class or category for a given observation. Classification can also be done using clustering analysis. Thus classification task can be performed using both Supervised and Unsupervised Machine Learning Techniques. The accuracy of the Classification problem is dependent on the type of dataset we have.If the distribution of the classes in the dataset is fairly even then Accuracy can be used as a parameter otherwise in case of imbalanced Classification problems we generally use Precision and Recall as metrics. Here in our problem we have computed the whole Classification Report which tells us all the three metrics.

2.1 SVM

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. To separate the two classes of data points, there are many possible hyperplanes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.



Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier.

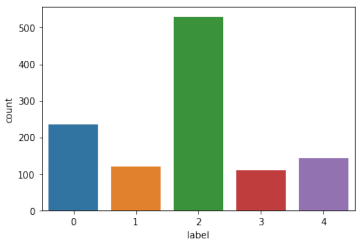
2.2 K Nearest Neighbour

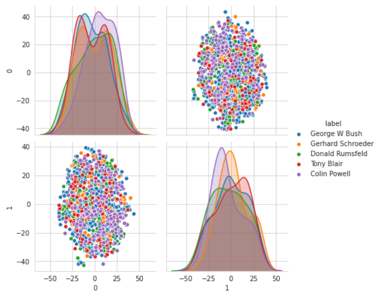
The principle behind nearest neighbor methods is to find a predefined number of training samples closest in distance to the new point, and predict the label from these. The number of samples can be a user-defined constant (k-nearest neighbor learning), or vary based on the local density of points (radius-based neighbor learning). The distance can, in general, be any metric measure: standard Euclidean distance is the most common choice. Neighbors-based methods are known as non-generalizing machine learning methods, since they simply "remember" all of its training data.

Despite its simplicity, nearest neighbors has been successful in a large number of classification and regression problems, including handwritten digits and satellite image scenes. Being a non-parametric method, it is often successful in classification situations where the decision boundary is very irregular.

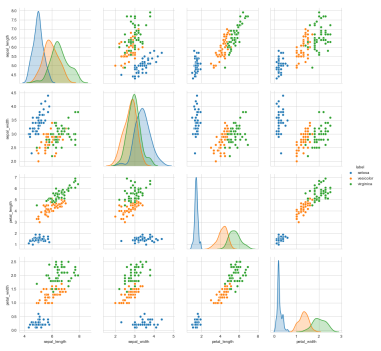
3 EDA

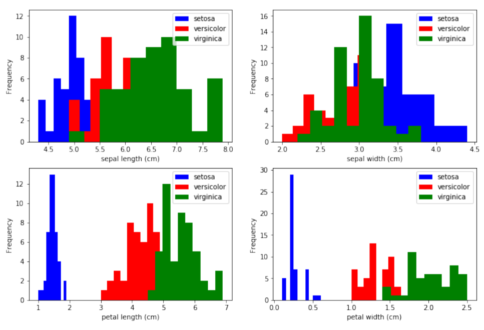
Exploratory Data Analysis is an important part of data science.Since doing this helps us understanding the data. The class wise distribution of faces in the wild dataset is as follows:





0:Colin Powell 1:Donald Rumsfeld 2:George W Bush 3:Gerhard Schroeder 4:Tony Blair. The above plot clearly states that the distribution in the dataset was not balanced and is highly overlapping. The iris dataset distribution is as follows:

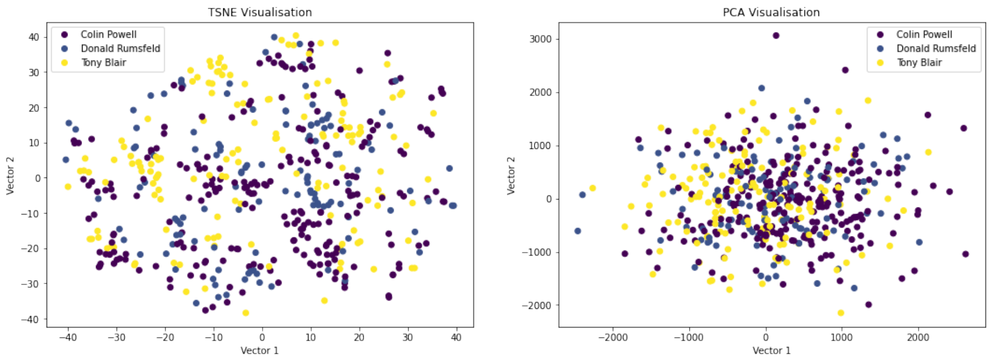




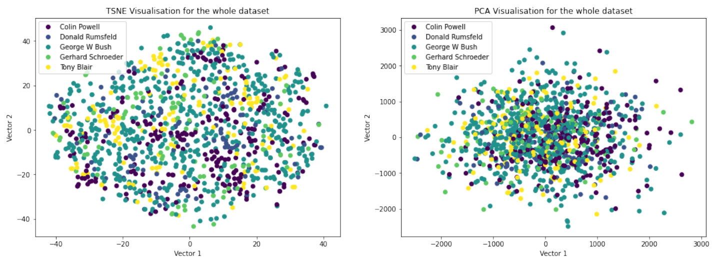
From the two above plots we can clearly observe that the sepal features do overlap for the three classes while the petal features are some what more distinct and could be more useful in classification.The number of examples for each class is equal for all the classes.

4 Question 1

The dataset used for this part of the assignment is the Faces in the Wild Dataset with only those celebrities images which have more than 100 samples(i.e. 5 as mentioned above along with their distribution).Eigenfaces is a method that is useful for face recognition and detection by determining the variance of faces in a collection of face images and use those variances to encode and decode a face in a machine learning way without the full information reducing computation and space complexity. Even though it was firstly used in 1991 by Turk and Pentland and has limitations compared to today's tech, its basics are still highly beneficial for new practices. In the Question we first transformed the images from 2914(62X47) dimensions to 100 dimensions using PCA.The total number of samples we had were 1194.So the dataset is essentially of the form 1140X100.We perform the train-test split (0.3 test-size). So after transforming all the images into 100Dimensions we select the Colin Powell,Tony Blair and Donald Rumsfield as the three identities and perform t-SNE to reduce their images into 2-D and view them.

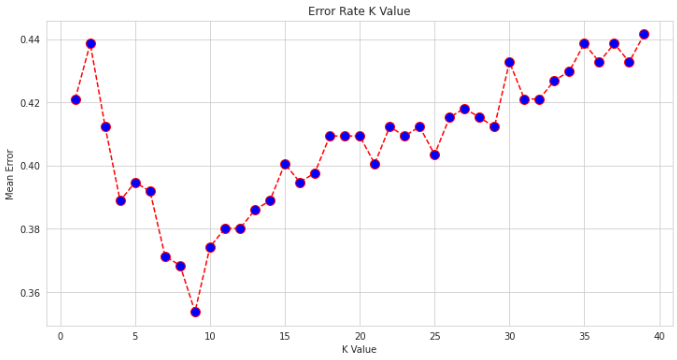


The images above show the t-SNE representation on the left and the PCA (using the first two eigen vectors) on the right of the three personalities.We can clearly observe that the clusters formed here are not absolutely differentiating clusters in either of the cases with the reasons for the observation could be the DATA ITSELF.All the three personalities(for that matter all the five) are **fair skinned men and all are aged belonging to the same age group** i.e. all the images almost belong to the same distribution as a whole. This causes most of the images being closely resembling each other which also gets embedded while performing t-SNE.We can observe that most of the images do appear in clusters but they are very small in size as compared to the total number of images of the class they belong,whereas in PCA all the images are clustered towards the centre since it does not work on increasing the class wise distribution.When we perform the t-SNE and PCA on the whole dataset, the result is:

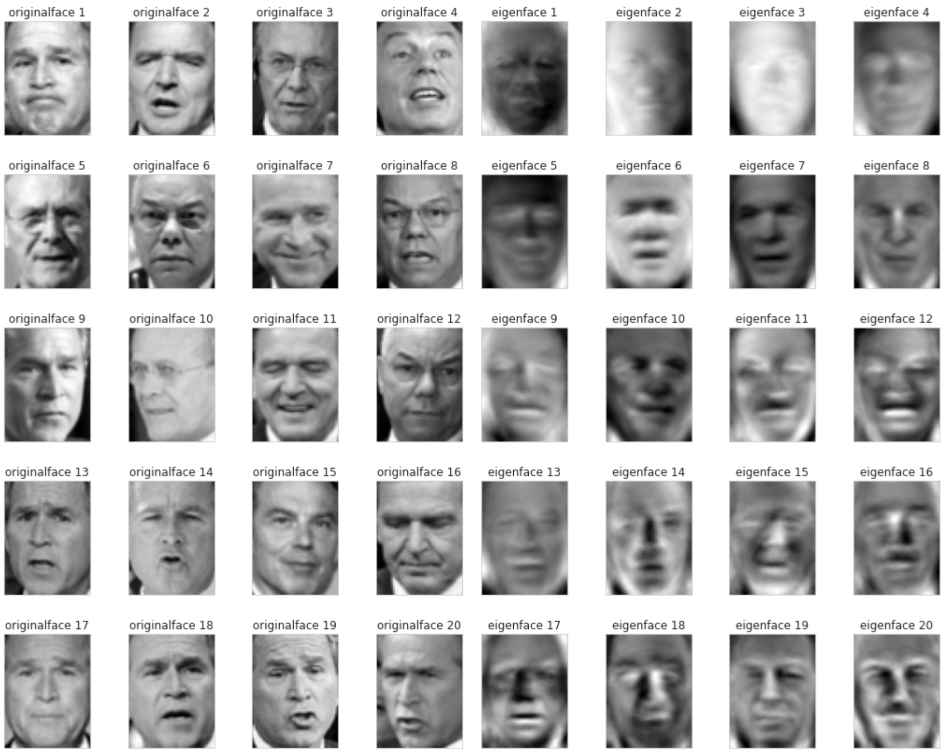


The overwhelming color green being present in the above two graphs is due to the fact that the number of images of George Bush in the dataset was way more than any other personality.The resemblance in the pixel values of the personalities which share several traits amongst themselves is the reason for the cluster being formed as such but still tiny clusters belonging to the same class are still visible as such.

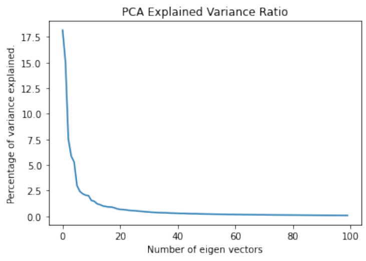
After observing the data via PCA and t-SNE we had to perform the classification using a Nearest Neighbour algorithm and generate classification report. To know the appropriate value of k I performed it for various value and recorded the error and chose the one with minimum which turned out to be equal to 9 in PCA's case and 55 in t-SNE.



RESULTS OF THE CLASSIFICATION ON THE KNN::
F1-Score:0.47 (for t-SNE) F1-Score:0.69 (for PCA)
Here are the first 20 eigen faces along with the corresponding real images:



The next part of the question asks to not hard code the number of PCA components and get 80 percent variance which turns out to be 31 first eigen vectors.The following graph shows the variance explained per PCA component.

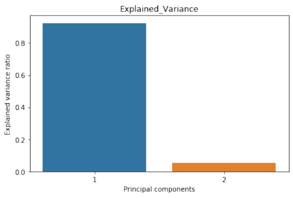


Thereafter again performing the KNN algorithm to get the results on the new dataset.This time the effective number of neighbours turns out to be 5. F1-Score:0.61(for 31 vector PCA)

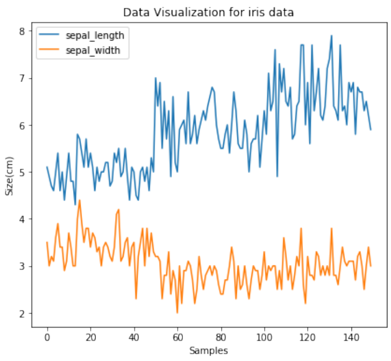
On comparing the above result with the one previously attained using 100 eigen vectors clearly shows that as we increase the number of eigen vectors we are saving more amount of information(variance) and thus have a better chance of getting a better result. So we get a better result in the previous case as compared to this one.

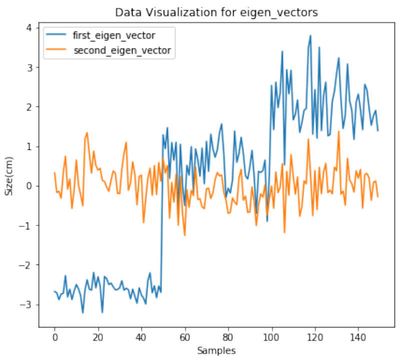
5 Question 2

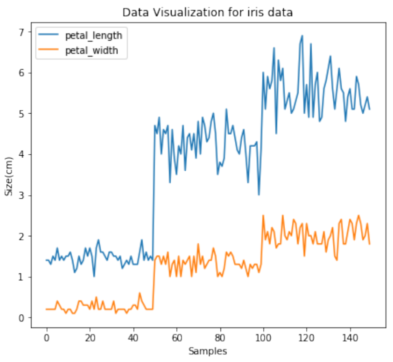
The second question uses the iris dataset where we perform various dimension reduction techniques such as t-SNE,PCA and LDA to reduce the dimensionality and visualize the data and try to understand the new visualisations of the data. First we use PCA to reduce the dimensions from 4 to 2.With the first eigen vector accounting for over 90 percent variance.

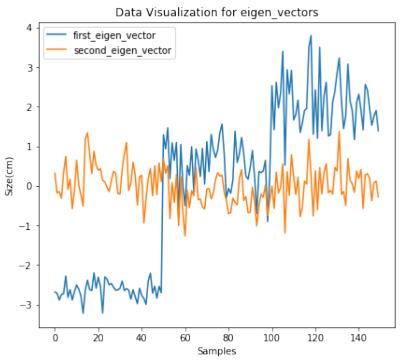


Then we also try to understand these two eigen vectors by figuring out what do they correspond to in the original feature space i.e high eigen value are in resemblance to which feature. The following two graphs help us in understanding it.

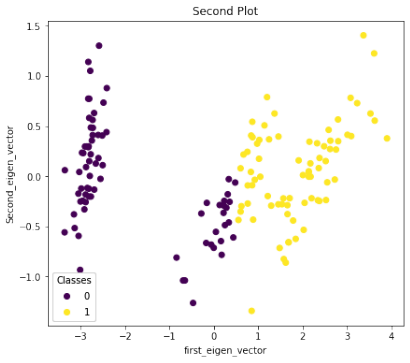


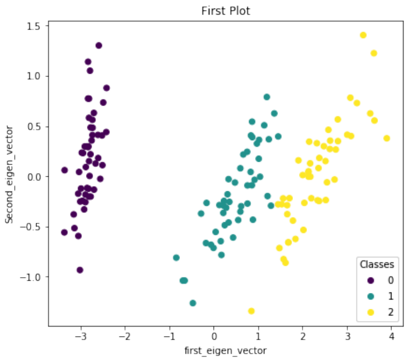


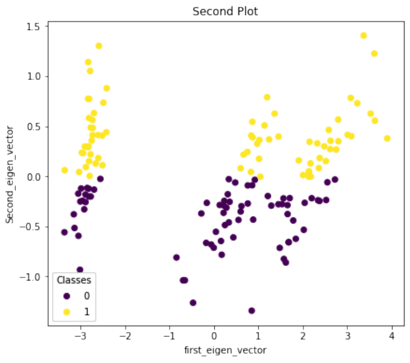


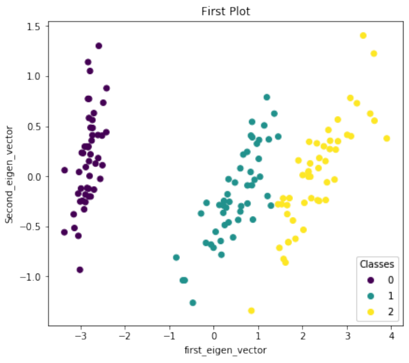


The first two graphs are corresponding to the sepal features on the left and the eigen vectors on the right for the 150 features(With the first 50 being setosa,next 50 being versicolor and the last 50 being virginica).**The graphs clearly show that the first the eigen vector clearly has lot of dependency on the petal length and petal width feature** as both show a very similar behaviour for all the three classes i.e. clearly increasing with each class. **The second eigen vector can be seen to show little resemblance to sepal width** feature as it is higher for the first class and relatively equal for the remaining two.**Sepal Length contributes little to none to both the eigen vectors.** Also on dividing the eigen vectors into 2 categories i.e. high and low and comparing it with the actual distribution of the three classes along these eigen vectors itself to check the eigen vector dependency.







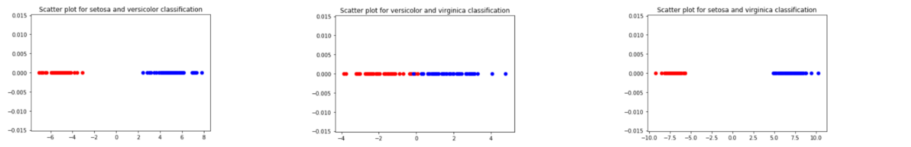


In the first two graph we can observe that the class 0 and 1 on the left graph relate to the mean of the first eigen vector that is if the value is higher than the mean it is labelled as class 1 or class higher otherwise as class 0 or class lower. If we compare the first two graphs we can easily relate that the lowest values of the first eigen vector corresponds to the class setosa in the the right graph, while the highest values of the first eigen vector corresponds to that of class virginica.Also the middle values or values close to the mean of the first eigen vector corresponds to the class versicolor.

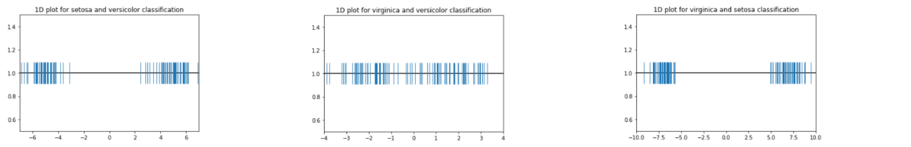
In the other two graphs the class 0 and 1 on the left graph relate to the mean of the second eigen vector similar to the one demonstrated in the first two graphs. If we compare the last two graphs it can be observed that approximately 50 percent of each setosa, virginica and versicolor classes are corresponding to the higher values of the second eigen vector.

So we conclude that the higher values of the first eigen vector correspond to the class virginica and lower values to class setosa and the mean values to class vesicular.Whereas we cannot draw any such conclusion for the second eigen vector as all the three classes almost have identical distribution across the second eigen vector distribution.So, the first eigen vector can serve significantly in classifying the three classes while the second eigen vector is of little or no use.

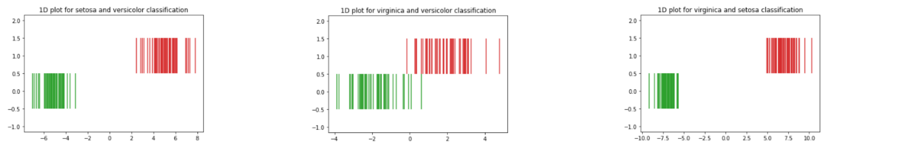
Now we perform Linear Discriminant Analysis to perform dimension reduction by considering only two classes at a time to a generate a total of 3 corresponding graphs of 1-D(LDA produces dimensions of size C-1 where C is the number of classes.) LDA can be seen doing pretty reasonable job in terms of setosa vs versicolor and setosa vs virginica class.The following are three 1-D graphs representations of the result of LDA.



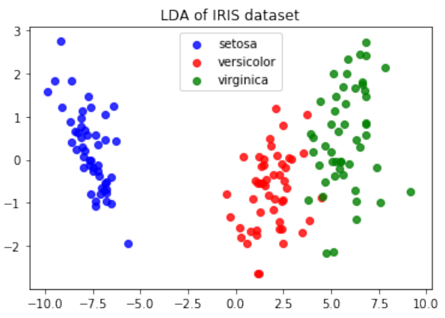
```
Threshold for setosa and versicolor classification: -5.08019541539774    Threshold for versicolor and virginica classification: -1.885396895083514    Threshold for setosa and virginica classification: -6.985440613150522
```

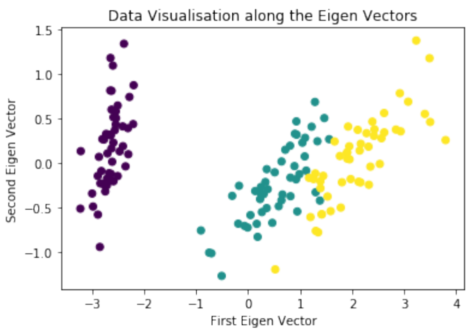


```
]: Text(0.5, 1.0, '1D plot for setosa and versicolor classification')    Text(0.5, 1.0, '1D plot for virginica and versicolor classification')    Text(0.5, 1.0, '1D plot for virginica and setosa classification')
```

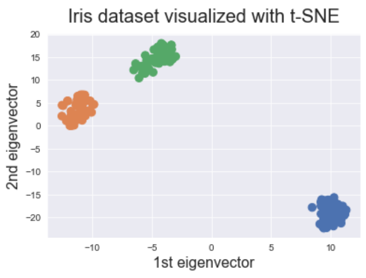


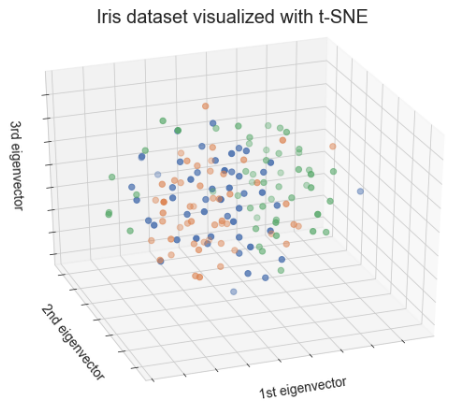
The following is the LDA on the complete dataset as opposed to PCA where clearly LDA does better in increasing inter class separation and decreasing intra class scatter.

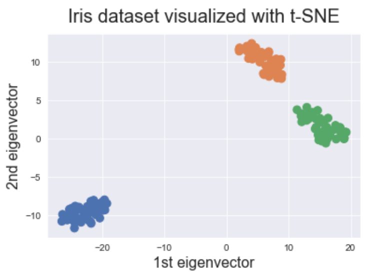


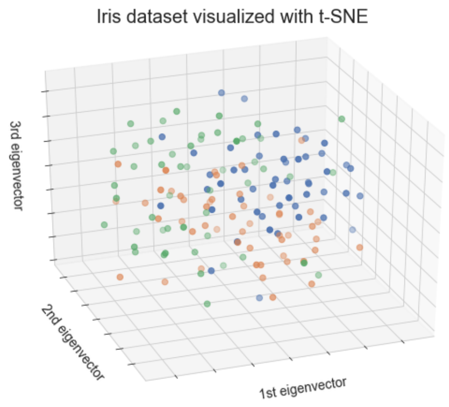


PCA can be used for capturing the global structure of the high-dimensional data but fails to describe the local structure within the data. Whereas, “t-SNE” is capable of capturing the local structure of the high-dimensional data very well while also revealing global structure such as the presence of clusters at several scales. The performance of t-SNE on the iris dataset by reducing it into 2-D and 3-D dataset is as follows:







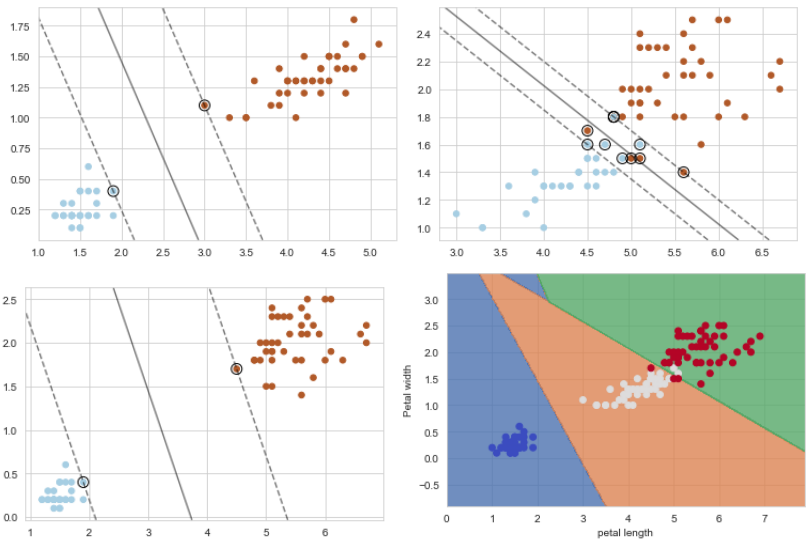


The metric used in the top two graphs is squeclidean while the metric used in the last two graphs is Manhattan.From the 2-D graphs the clustering of the classes under both the metrics is clearly visible with both the metrics producing reasonably good results.The clustering is a bit more complete in the case of squeclidean metric case. The three classes can be easily viewed to be quite separated in the case of the 2-D plot but in the case of the 3-D plot it is quite difficult to analyse the same but on inspecting the values manually one can clearly see that the algorithm does a better job in case of squeclidean.But to straight away draw conclusions will be tough from just viewing a 3-D plot in 2-D form.

6 Question 3

In the last question we have to focus on the usage of Support Vector Machine for classification of the iris dataset using only the petal features of the iris dataset. Here are the results of using the linear kernel on the iris dataset. First the SVM has been used to perform classification on pair wise classes.

F1-Score: 1.00 F1-Score: 0.97 F1-Score: 1.00 F1-Score: 0.95 (for the complete dataset) Here are the graphs:



The first three graphs depict the pair wise class classification of the iris dataset whereas the fourth graph shows the complete classification of all the three categories along with the hyperplane and the support vectors for the graphs.In the first and the third graph the support vectors are far away from the central hyperplane showing that the classes have clearly distinct feature values. While in the second graph the number of misclassified points is higher and the width of the hyperplane is less showing an overlap between the feature values. Now on changing the value of the C parameter in the above situations the accuracy are as follows:

C=0.001 F1-Score: 0.30 F1-Score: 0.30 F1-Score: 0.30 F1-Score: 0.20 (for the complete dataset)

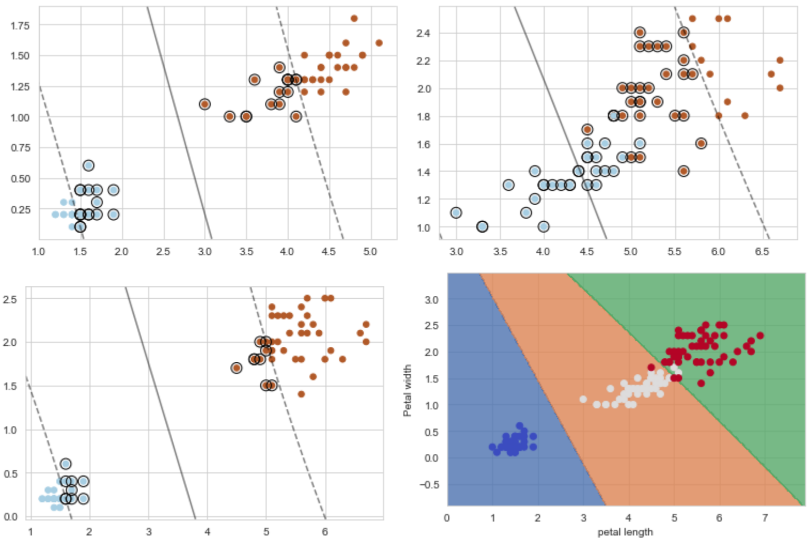
C=1000 F1-Score: 1.0 F1-Score: 0.96 F1-Score: 1.0 F1-Score: 0.93 (for the complete dataset)

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, we should get misclassified examples, often even if your training data is linearly separable.This is what happens with above mentioned F1-Score which are significantly lower for C=0.001 while good enough for C=1000. (In layman terms it is the inverse of the regularization parameter so lower the value of the C parameter more strict and severe the regularization shall be while larger value of C shall perform better).

In the last part of the assignment we use the RBF kernel to perform the classification. $K(\mathbf{x},\mathbf{x}') = \exp\left(-\frac{||\mathbf{x}-\mathbf{x}'||^2}{2\sigma^2}\right)$

where the numerator in the exponential is the squared Euclidean distance between two data points x and x'.It is important to know that an SVC classifier using an RBF kernel has two parameters: gamma and C. Gamma is a parameter of the RBF kernel and can be thought of as the 'spread' of the kernel and therefore the decision region. When gamma is low, the 'curve' of the decision boundary is very low and thus the decision region is very broad. C is a parameter of the SVC learner and is the penalty for misclassifying a data point. When C is small, the classifier is okay with misclassified data points (high bias, low variance). When C is large, the classifier is heavily penalized for misclassified data and therefore bends over backwards avoid any misclassified data points (low bias, high variance).

The plots are as follows :



Here the number of misclassified samples is higher as compared to linear kernel. From the graphs it is quite clear the performance of the SVM with the linear kernel doing a pretty reasonable job as compared to the RBF kernel.With the sole reason being the fact that using the petal features the classes are linearly separable and do not require a complex function such as the RBF kernel for classification which leads to the overfitting of the train dataset.But when C=0.001 the performance is identical for both the kernels because of the harsh penalty and in case of C=1000 as well the performance is almost identical with the sole objective being to achieve more and more accuracy. The F1-Score are as follows:

C=0.001 F1-Score: 0.30 F1-Score: 0.30 F1-Score: 0.30 F1-Score: 0.20 (for the complete dataset)

c=1 F1-Score: 1.00 F1-Score: 0.73 F1-Score: 1.00 F1-Score: 0.91 (for the complete dataset)

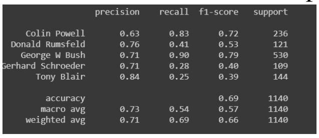
C=1000 F1-Score: 1.0 F1-Score: 0.97 F1-Score: 1.0 F1-Score: 0.91 (for the complete dataset)

7 Conclusions and Learnings

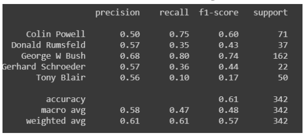
The main learning from this assignment is the usage of the dimensionality reduction and its importance.When we have labels for the data then it is advisable to use LDA since it provides accurate mapping from higher dimension.When we don't have data and want to significantly reduce the dimensions independent of the number of classes then PCA and t-SNE are better. The t-SNE algorithm is stochastic whereas PCA is deterministic meaning it will always give the same directions and can explain as well.

Only a first few eigen vectors are of significant importance in case of PCA.The C parameter in the case of SVM has a very strong impact on the accuracy of the algorithm and if the data is linearly separable then there is not much benefit of using a non linear kernel.

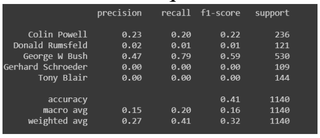
Here are the classification reports for the KNN algorithm used in the first question.



```
                  precision    recall  f1-score   support

     Colin Powell       0.63      0.83      0.72       236
  Donald Rumsfeld       0.76      0.41      0.53       121
    George W Bush       0.71      0.90      0.79       530
erhard Schroeder       0.71      0.28      0.40       109
       Tony Blair       0.84      0.25      0.39       144

         accuracy                           0.69      1140
        macro avg       0.73      0.54      0.57      1140
     weighted avg       0.71      0.69      0.66      1140
```

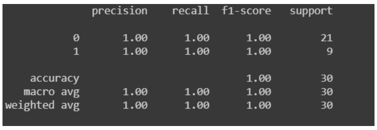


```
                  precision    recall  f1-score   support

     Colin Powell       0.50      0.75      0.60        71
  Donald Rumsfeld       0.57      0.35      0.43        37
    George W Bush       0.68      0.80      0.74       162
erhard Schroeder       0.57      0.36      0.44        22
       Tony Blair       0.56      0.10      0.17        50

         accuracy                           0.61       342
        macro avg       0.58      0.47      0.48       342
     weighted avg       0.61      0.61      0.57       342
```

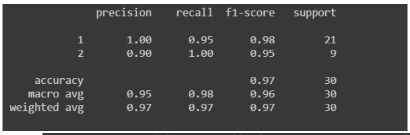


```
                  precision    recall  f1-score   support

     Colin Powell       0.23      0.20      0.22       236
  Donald Rumsfeld       0.02      0.01      0.01       121
    George W Bush       0.47      0.79      0.59       530
Gerhard Schroeder       0.00      0.00      0.00       109
       Tony Blair       0.00      0.00      0.00       144

         accuracy                           0.41      1140
        macro avg       0.15      0.20      0.16      1140
     weighted avg       0.27      0.41      0.32      1140
```

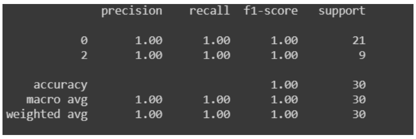
Here are the classification report for the SVM algorithm with linear kernel and C=1 (first four) C=0.001 and C=1000 next four.



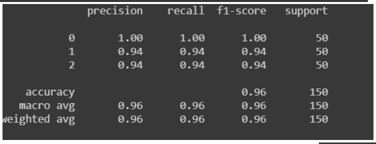
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        21
           1       1.00      1.00      1.00         9

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```



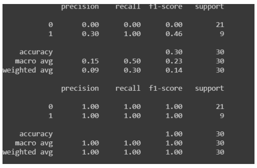
```
              precision    recall  f1-score   support

           1       1.00      0.95      0.98        21
           2       0.90      1.00      0.95         9

    accuracy                           0.97        30
   macro avg       0.95      0.98      0.96        30
weighted avg       0.97      0.97      0.97        30
```



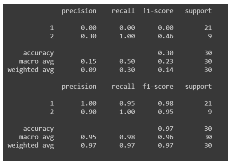
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        21
           2       1.00      1.00      1.00         9

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```



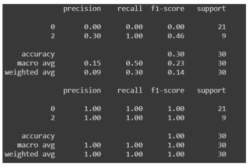
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        50
           1       0.94      0.94      0.94        50
           2       0.94      0.94      0.94        50

    accuracy                           0.96       150
   macro avg       0.96      0.96      0.96       150
eighted avg       0.96      0.96      0.96       150
```



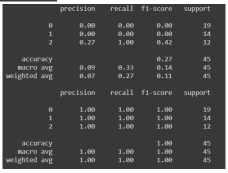
```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        21
           1       0.30      1.00      0.46         9

    accuracy                           0.30        30
   macro avg       0.15      0.50      0.23        30
weighted avg       0.09      0.30      0.14        30
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        21
           1       1.00      1.00      1.00         9

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```



```
              precision    recall  f1-score   support

           1       0.00      0.00      0.00        21
           2       0.30      1.00      0.46         9

    accuracy                           0.30        30
   macro avg       0.15      0.50      0.23        30
weighted avg       0.09      0.30      0.14        30
              precision    recall  f1-score   support

           1       1.00      0.95      0.98        21
           2       0.90      1.00      0.95         9

    accuracy                           0.97        30
   macro avg       0.95      0.98      0.96        30
weighted avg       0.97      0.97      0.97        30
```

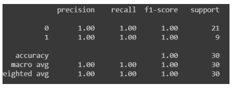


```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        21
           2       0.30      1.00      0.46         9

    accuracy                           0.30        30
   macro avg       0.15      0.50      0.23        30
weighted avg       0.09      0.30      0.14        30
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        21
           2       1.00      1.00      1.00         9

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

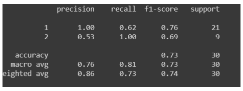


```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        19
           1       0.00      0.00      0.00        14
           2       0.27      1.00      0.42        12

    accuracy                           0.27        45
   macro avg       0.09      0.33      0.14        45
weighted avg       0.07      0.27      0.11        45
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        19
           1       1.00      1.00      1.00        14
           2       1.00      1.00      1.00        12

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
weighted avg       1.00      1.00      1.00        45
```

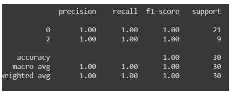
Here are the classification report for the SVM algorithm with rbf kernel and C=1(first four) c=0.001 and c=1000 next four.



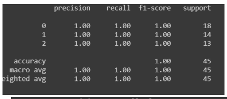
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        21
           1       1.00      1.00      1.00         9

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
eighted avg       1.00      1.00      1.00        30
```



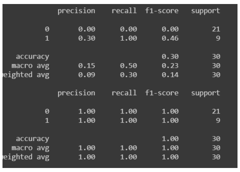
```
              precision    recall  f1-score   support

           1       1.00      0.62      0.76        21
           2       0.53      1.00      0.69         9

    accuracy                           0.73        30
   macro avg       0.76      0.81      0.73        30
eighted avg       0.86      0.73      0.74        30
```



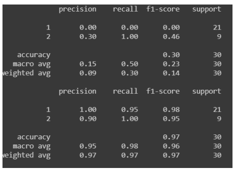
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        21
           2       1.00      1.00      1.00         9

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
eighted avg       1.00      1.00      1.00        30
```



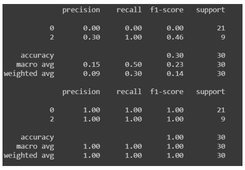
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        18
           1       1.00      1.00      1.00        14
           2       1.00      1.00      1.00        13

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
eighted avg       1.00      1.00      1.00        45
```



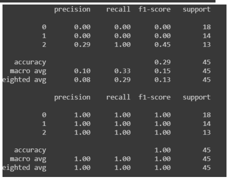
```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        21
           1       0.30      1.00      0.46         9

    accuracy                           0.30        30
   macro avg       0.15      0.50      0.23        30
eighted avg       0.09      0.30      0.14        30
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        21
           1       1.00      1.00      1.00         9

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
eighted avg       1.00      1.00      1.00        30
```



```
              precision    recall  f1-score   support

           1       0.00      0.00      0.00        21
           2       0.30      1.00      0.46         9

    accuracy                           0.30        30
   macro avg       0.15      0.50      0.23        30
eighted avg       0.09      0.30      0.14        30
              precision    recall  f1-score   support

           1       1.00      0.95      0.98        21
           2       0.90      1.00      0.95         9

    accuracy                           0.97        30
   macro avg       0.95      0.98      0.96        30
eighted avg       0.97      0.97      0.97        30
```



```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        21
           2       0.30      1.00      0.46         9

    accuracy                           0.30        30
   macro avg       0.15      0.50      0.23        30
weighted avg       0.09      0.30      0.14        30
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        21
           2       1.00      1.00      1.00         9

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```



```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        18
           1       0.00      0.00      0.00        14
           2       0.29      1.00      0.45        13

    accuracy                           0.29        45
   macro avg       0.10      0.33      0.15        45
eighted avg       0.08      0.29      0.13        45
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        18
           1       1.00      1.00      1.00        14
           2       1.00      1.00      1.00        13

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
eighted avg       1.00      1.00      1.00        45
```