**Letter Recognition**

***PCA (Unsupervised Dimensionality Reduction Technique)***

The central idea of principal component analysis (PCA) is to reduce the dimensionality of a data set consisting of a large number of interrelated variables while retaining as much as possible of the variation present in the data set. This is achieved by transforming to a new set of variables, the principal components (PCs), which are orthogonal , and which are ordered so that the first few retain most of the variance present in all of the dataset.

PCA can be thought of as an unsupervised learning problem. The whole process of obtaining principal components from a raw dataset can be simplified in six parts :

1. **Take the whole dataset consisting of d+1 dimensions and ignore the labels such that our new dataset becomes d dimensional**. It is important as we cannot transform the class labels.
2. **Compute the mean of the dataset across each column and subtract it from every column**:- Centering the dataset is one the most important step the reason is that it becomes computationally efficient to calculate covariance matrix.
3. **Compute the covariance matrix of the whole dataset:-**If the dataset is centered from each column so Covariance Matrix can be calculated directly using **(X.T.X)/m**  as all the data.(here T represents Transpose.



1. **Compute eigenvectors and the corresponding eigenvalues**:-Let A be a square matrix, ν a vector and λ a scalar that satisfies Aν = λν, then λ is called eigenvalue associated with eigenvector ν of A. Eigenvalues can be calculated by **det(A-λI)=0**.
2. **Sort the eigenvectors by decreasing eigenvalues and choose k eigenvectors** with the largest eigenvalues to form a d × k dimensional matrix W.
3. **Use this d × k eigenvector matrix to transform the samples onto the new subspace.**y = W′ × x where W′ is the transpose of the matrix W.

***LDA (Supervised Dimensionality Reduction Technique)***

LDA is a supervised data compression technique which is aimed at increasing class distinction techniques. The goal in LDA is to find the feature subspace that optimizes class separability and to serve this purpose it requires the class labels.

Steps involved:

1. Standardise the d-dimensional dataset.

* Standardising the data-set is important before computing the LDA otherwise, it would give us wrong answers. It is required so as to bring the entire data onto the same scale for getting comparable results.

2. For each class compute the d-dimension mean vectors.

* The next step is finding the mean of each column grouped by their classes of output. This step actually allows us to create a vector data for class separation in LDA.

3. Construct between class-scatter matrix and within class-scatter matrix.

* The next step is the calculation of within class scatter-matrix and the between class scatter matrix i.e. to reduce the distance between two classes we need to reduce the variation within a class and increase the variation among different classes.
* It is computed using the formula given below:

Here, Si denotes the variance of the data for any particular class, and Sw is the within class scatter matrix and is the sum of the variances over all the classes. We take the sum as we want to minimize the overall sum of all the variances together rather than minimizing the variance for a single class.

* We next find out the between class scatter-matrix. It is computed using the formula given below:



This gives us the variation in the means of our data between different classes. Here m denotes the overall mean including all the classes.

4. Compute eigenvector for the matrix W displayed below.

* Next, we compute the eigenvectors of the matrix W given below:



Here Sw and Sb are within class and between class scatter matrix. We choose the matrix specifically because this function basically tends to minimize the within class scatter and maximize the between class scatter. The reason for taking eigenvectors is to map the feature space into a newer subspace.

*5.* Choose the k eigen-vectors which correspond to k-largest eigen-values to construct a (d,k) dimensional transformation matrix W (the eigen-vectors are the columns of this matrix).

6. Project X on matrix W i.e. space -transformation.

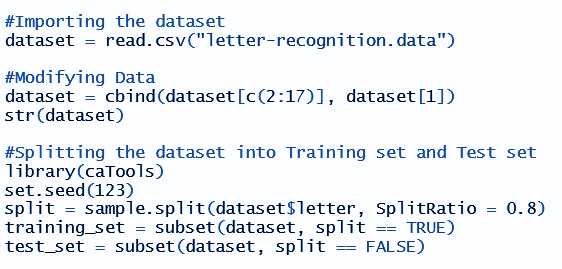
* The steps 5 and 6 create a new matrix, say, M using only the first k- eigenvectors and using these vectors to convert the data into a new feature-set.

***PCA & LDA implementation in R:***

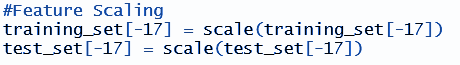
We have applied PCA & LDA on Letter Recognition dataset

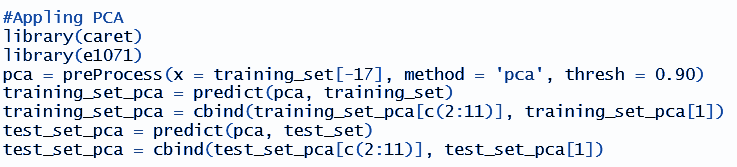
Process:

* First, we import and pre-process the data. Then we split the data into training and testing using the same seed, so that we can compare the results of both the methods.

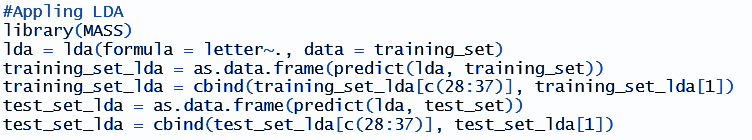


* Then we standardize both the training and testing data.

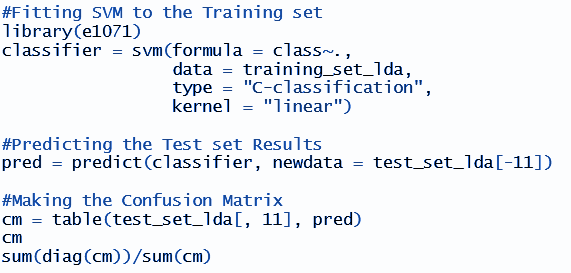


* ***PCA*** – Then we train a model on training data called ‘pca’ using the ‘preProcess’ function to get principal components with 90% variance of the original data. Then we predict principal components for both training and testing data using the pca model.

***LDA*** – Then we train a model on training data called ‘lda’ using the ‘lda’ function. Then we predict linear discriminants for both training and testing data using the lda model and disregard all the LDs other than the 10 most effective LDs.



* Then we use the ‘svm’ function to model a classifier which will be used in predicting the class of test data.



* After plotting the confusion matrix, we get the accuracy of the following models as follows:

Accuracy with PCA – 76.13%

Accuracy with LDA – 89.75%

* ***PCA***– Then we plot the training data along PC1 and PC2 to visualise that PC1 is able to explain 26.87% variance of the data and PC2 is able to explain 16.33% variance of the data.

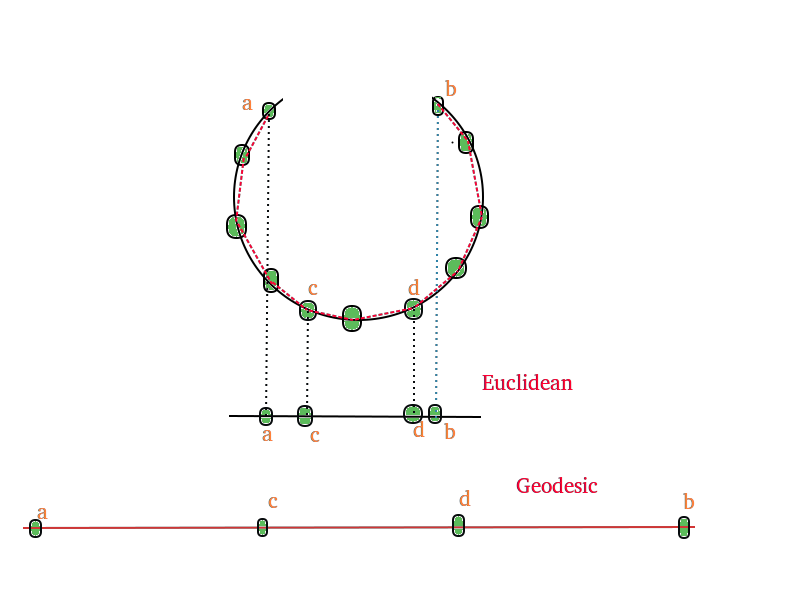
***LDA*** *–* Then we plot the training data along LD1 and LD2 to visualise that we got a separability of 31.25% along LD1 and 21.29% along LD2.

***Both the plots are given in (/Letter Recognition/Plots/)***

**Breast Cancer Wisconsin**

***Isomap***

Isomap stands for isometric mapping. It is a form of metric learning. Isomap is a non-linear dimensionality reduction method which tries to preserve the geodesic distances in the lower dimension. Isomap starts by creating a neighborhood network. After that, it uses graph distance to the approximate geodesic distance between all pairs of points. And then, through eigenvalue decomposition of the geodesic distance matrix, it finds the low dimensional embedding of the dataset.The geodesic distances between two points in the image are approximated by graph distance between the two points. Thus, Euclidean distances should not be used for approximating the distance between two points in nonlinear manifolds while geodesic distances can be used.

The following image shows Euclidean and geodesic distances for a set of points on a circular manifold.

Isomap uses the above principle to create a similarity matrix for eigenvalue decomposition. Unlike other non-linear dimensionality reduction like LLE & LPP which only use local information, isomap uses the local information to create a global similarity matrix. The isomap algorithm uses euclidean metrics to prepare the neighborhood graph. Then, it approximates the geodesic distance between two points by measuring the shortest path between these points using graph distance. Thus, it approximates both global as well as the local structure of the dataset in the low dimensional embedding.

***Steps***:

Isomap differs from classical Multidimensional Scale in initial few steps only. Instead of using euclidean metric for dissimilarity, it uses graph distances. Steps of the Isomap algorithm are:

1. Neighbourhood graph
2. Dissimilarity Matrix
3. Eigenvalue decomposition

Firstly it determines which points are neighbors on Manifold based on distance (Euclidean distance). For each point, it connects all points within a fixed radius where it does so by choosing radius or by fixing the number of nearest neighbours to take. All neighborhood relations are represented as a weighted graph.

Next it estimates the geodesic distance between all pairs of points. To measure the shortest distance, it moves along the shortest path.

In the next step, it applies MDS (Multidimensional Scaling) to the matrix of graph distances due to constructing an embedding of the data in a d-dimensional space (Euclidean space) where the manifold’s geometry is preserved. The coordinate vectors in the Euclidean-Space are chosen due to minimizing the cost function:(where Dy is the Euclidean-based matrix of distances and L2 corresponds to the square of the sum of elements. **τ** is a function that converts distances to inner products (due to support efficient optimization)). The global minimum is achieved by setting Euclidean-space’s coordinates to the top **d** eigenvectors of the matrix.

***Algorithm:***

The Isomap algorithm projects a finite subset of points X Rn to RK (for some positive

given K < n) as follows:

1. It computes all pairwise distances for X, yielding the distance matrix D

2. It selects a subset d of “short” Euclidean distances in D (usually up to a given threshold),

yielding a simple connected weighted graph G = (V,E,d) where d : E R+;

3. It computes all shortest paths in G, and produces an approximate distance matrix ,

where ij = dij for all {i, j} E and ij is the value of the shortest path from i to j

otherwise;

4. It derives a corresponding approximate Gram matrix by setting

= J2J

where J = In ≠ 11T;

5. It finds the (diagonal) eigenvalue matrix of and the corresponding eigenvector matrix

P, so that = PTP;

6. Since is only an approximation of a Gram matrix, it might have some negative

eigenvalues: Isomap replaces all the negative eigenvalues with zeroes;

7. In case there are still more than K positive eigenvalues, Isomap replaces the smallest

ones, leaving only the largest K eigenvalues on the diagonal of a PSD matrix ;

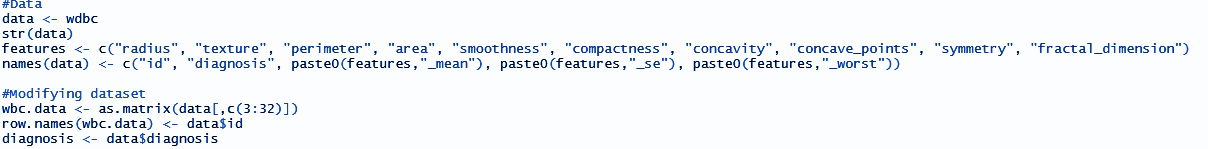
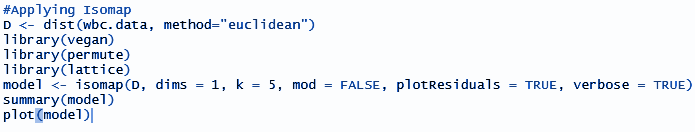
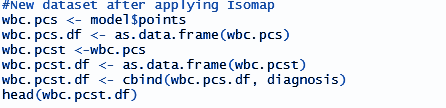
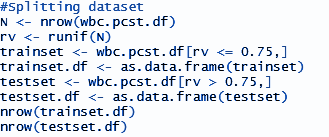
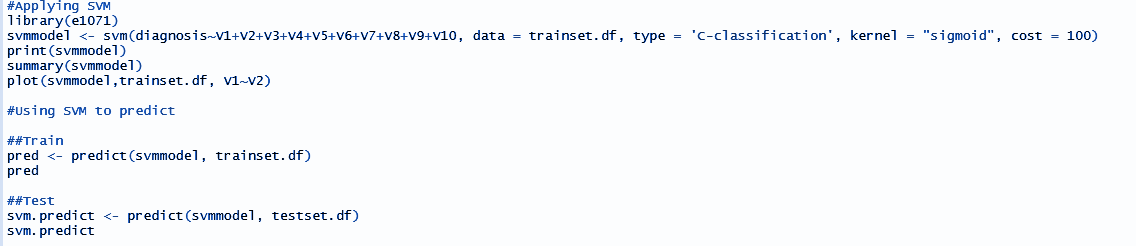
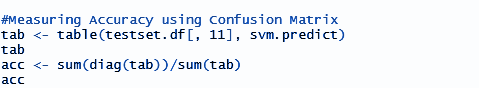
8. Finally, it sets x = PT.

(Note: represents D bar and similarly for )

***Isomap implementation in R:***

We have applied the Isomap algorithm on Breast Cancer (diagnostic) dataset:

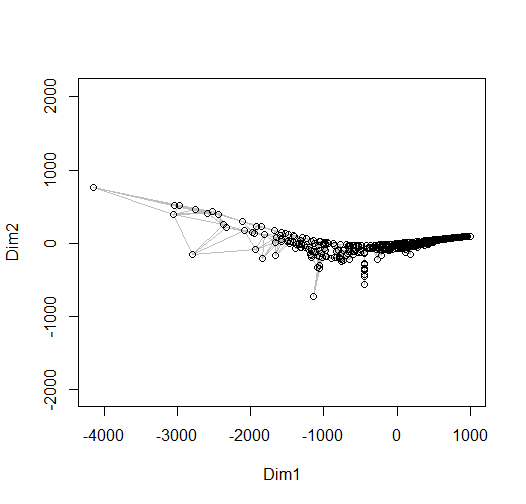
Process:

* First we import and store the dataset. We add to it the column names of the different variables. We then also store the data other than the response variable and a column vector that has the response variable values.  
    
  
* ***Isomap:*** We train the Isomap model. First we calculate the distance matrix, then we apply isomap on this matrix.  
    
  
* We store the new dataset with reduced dimensions and the corresponding values of the response variable to each row.   
    
  
* We split the dataset into train and test sets  
    
  
* We use the SVM function to model a classifier. We use this classifier to predict values of the response variable for the test dataset.  
    
    
  
* We find out the accuracy of our classifier by tabulating values of the actual and the predicted classes using a Confusion Matrix and calculate the value of accuracy.  
    
  

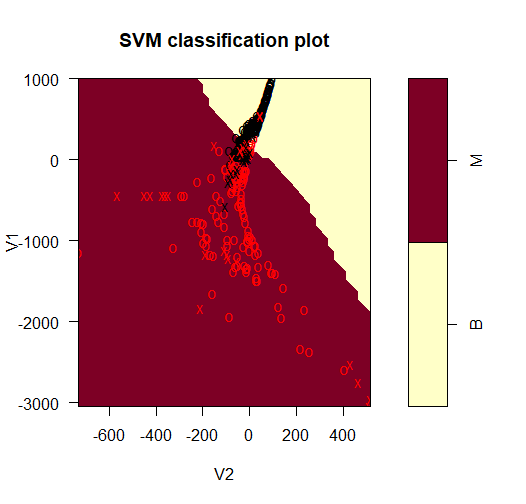
Accuracy using Isomap algorithm = 91.89189%

We have two plots:

Plot of dimension 1 vs dimension 2 after isomap has been applied:



Plot of the Support Vector Machine model for the first two variables:



**t-SNE (T-dist stochastic neighbour embedding):**

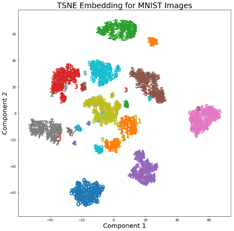
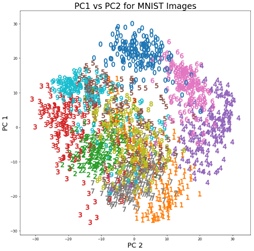
One of the most powerful and state of the art techniques is called t-SNE as it is one of the most important techniques which check “similarity” of points around using the Euclidean metric and check how points correlate to each and use this feature to reduce the dimension of data. It is also a great technique to visualize the data.

Geometric intuition of t-SNE implies that t-SNE as an idea tries to find a direction that preserve the neighbourhood (all the points whose Euclidean distance in the given metric space tends to be small) in N dimensional space to neighbourhood of points in projected or reduced dimensional space.

Since it preserves the neighbourhood of points in reduced dimension space also so points which are separated in N dimensional space are also separated in reduced dimensional space hence it is used for clustering also.

Here Embedding refers to shifting from a higher dimensionality space to a lower dimensionality space.

Main problem of Principal Component analysis is that it looks at the global structure or global variance preservation but t-SNE looks for local preservation of structure that is why t-SNE is much more better data visualization algorithm than PCA which does not preserve the structure of the data.



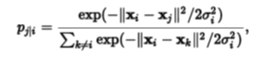
In t-SNE what we do is calculate the distance of a point from all the remaining points .

The point with high “similarity”- in original dimensional space should be kept together so we calculate the similarity scores using t-distribution as we measure the Euclidean distance from all the means considering the point from which all distance are measured to be as mean point(or center) of t-distribution we then calculate the similarity scores corresponding to the values in t-distribution.

We find the direction that minimizes the ratio of similarity in original dimensional space to reduced dimensional space.

In a more mathematical way it is represented as:-

given a set of N high dimensionality objects ,x1,x2,……xn.it calculates the value of



The similarity of datapoint to datapoint is the conditional probability, , that would pick as its neighbour if neighbours were picked in proportion to their probability density under a Gaussian centred at ."



Moreover, the probabilities with  are set to zero:

i = j

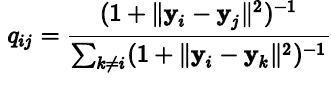
the density of the data: smaller values of are used in denser parts of the data space

Since the Gaussian kernel uses the Euclidean distance  , it is affected by the curse of dimensionality, and in high dimensional data when distances lose the ability to discriminate, the  become too similar (asymptotically, they would converge to a constant). It has been proposed to adjust the distances with a power transform, based on the intrinsic dimension of each point, to alleviate this.[12

It is really important for us to choose a proper metric distance so that we can get better performance.

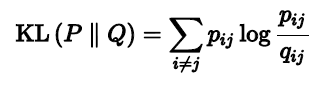
t-SNE aims to learn a d -dimensional map (with ) that reflects the similarities as

well as possible. To this end, it measures similarities qij between two points in the map yi and yj , using a very similar approach. Specifically,qij is defined as:



Herein a heavy-tailed Student t-distribution (with one-degree of freedom, which is the same as a Cauchydistribution) is used to measure similarities between low-dimensional points in order to allow dissimilar objects to be modeled far apart in the map. Note that also in this case we set qii=0

The locations of the points yi in the map are determined by minimizing the (non-symmetric) Kullback–Leibler divergence of the distribution P from the distribution Q , that is:



The minimization of the Kullback–Leibler divergence with respect to the points yi is performed using gradient descent. The result of this optimization is a map that reflects the similarities between the high dimensional inputs well.

**perplexity**:-Perplexity is a measure for information that is defined as 2 to the power of the Shannon entropy. The perplexity of a fair die with k sides is equal to k. In t-SNE, the perplexity may be viewed as a knob that sets the number of effective nearest neighbors. It is comparable with the number of nearest neighbors k that is employed in many manifold learners.The most appropriate value depends on the density of your data. **Loosely speaking, one could say that a larger / denser dataset requires a larger perplexity.** Typical values for the perplexity range between 5 and 50.

Another intuition for t-SNE is from how manifold learning works. A Manifold is a dd-dimensional surface that lives in an DD-dimensional space, where d<Dd<D. For the 3D case, imagine a 2D piece of paper that is embedded within 3D space. Even if the piece of paper is crumpled up extensively, it can still be ‘unwrapped’ (uncrumpled) into the 2D plane that it is. This 2D piece of paper is a manifold in 3D space. Or think of an entangled string in 3D space – this is a 1D manifold in 3D space.

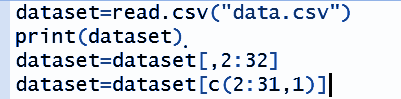
Now there’s what is known as the Manifold Hypothesis, or the >Manifold Assumption, that states that natural data (like images, etc.) forms lower dimensional manifolds in their embedding space. If this assumption holds (there are theoretical and experimental evidence for this hypothesis), then t-SNE should be able to find this lower-dimensional manifold, ‘unwrap it’, and present it to us as a lower-dimensional map of the original data.

***t-SNE and PCA implementation in R****:*

We have applied t-SNE And PCA on Breast Cancer (diagnostic)dataset.

***PCA***

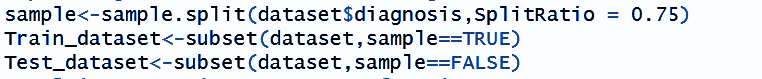
* Import the dataset using read.csv and setting the dataset so that labels are at the last columns



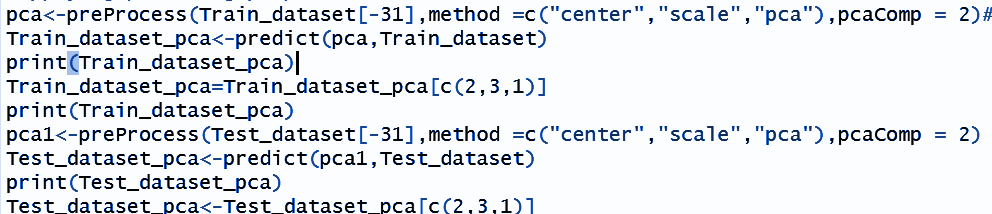
* Encoding labels (“M” and “B”) into numeric (0,1) using factor function



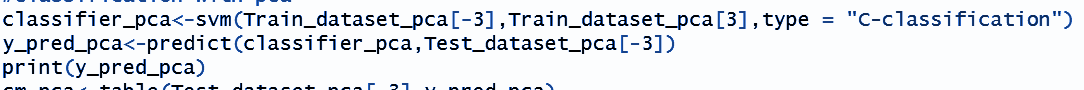
* Splitting the dataset into Training and Testing .



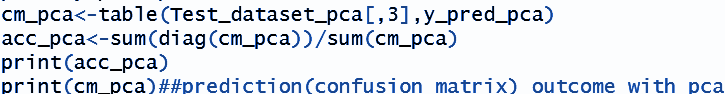
* Applying PCA using Preprocess function:-



* After Transforming the dataset applying classification using svm with a prebuilt in function in R called svm .

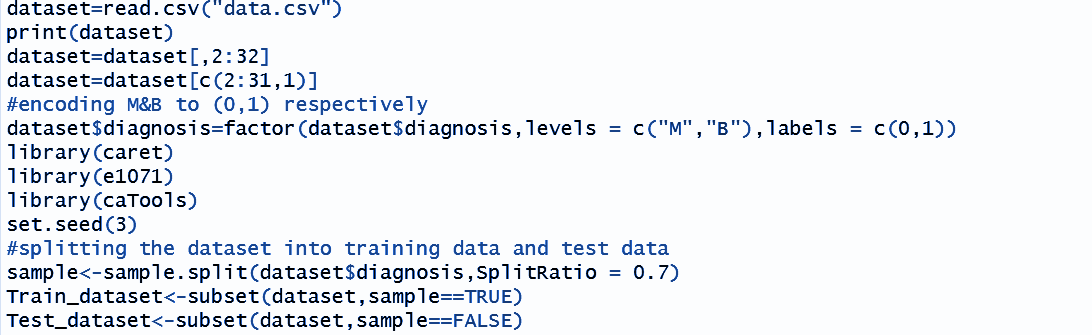


* Then accuracy was obtained from confusion matrix(command function in R:-table())by summing up the diagonal values of the confusion matrix and dividing it by Total

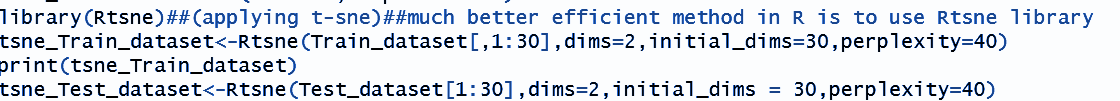


***t-SNE***

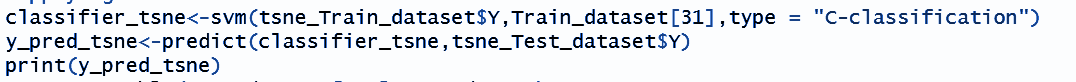
* Doing similarly like earlier we prepocess dataset ,**encode it and split into train and test set.**



* Now applying t-SNE in R using R built in a much more efficient function for t-SNE which is the Rtsne function and transforming our dataset.



* Now applying classification using svm classifier,



* Now checking the accuracy:-



At end we compare the results :-

Accuracy without using any dimensionality reduction - 97.06%

Accuracy with using PCA(30 dimensions ->2dimensions) - 92.23%

Accuracy with t-SNE reducing (30 dimensions ->2 dimensions) - 78.94%

**Group** – Patel Darsh Rajesh 2018B4A70532P

Manan Malani 2018B4A70709P

Anuraj Som 2018A1PS0037P