SMAI(Mini Project 2) Report

By: Aditya Agarwal

Roll No: 20161104

For this mini project we have used the cifar-10 dataset. First, we will describe the dataset:

- The dataset is broken into batches, namely data_batch_1, data_batch_2 and so on.

- The whole of the batched is of 10000*3072 dimensions, which means that it consists of 10000 images and each image is of 3072 dimensions.

- The labels consists of labels from 0 to 9 and thus we can say that the number of classes = 10.

- The classes namely are:
  - Airplane
  - Automobile
  - Bird
  - Cat
  - Deer
  - Dog
  - Frog

- Horse
- Ship
- Truck

- We then load the dataset and then use 80% of it for training and the rest 20% of it for testing.

PREPROCESSING THE DATASET:

- We have preprocessed the dataset in order to reduce the computation complexity as well as to reduce the time of the computation of results.
  - Bird: 1032
  - Cat: 1016
  - Deer: 999
  - Dog: 937
  - Frog: 1030
  - Horse: 1001
  - Ship: 1025
  - Truck: 981

- We have used scaling to scale the data, scaling basically standardizes the dataset along any axes which is defined in the argument.

● This normalises the data images which thus helps in reducing the computation.

● We are thus scaling and transforming the given test data and the training dataset as well.

PROCEDURE FOLLOWED

- We have selected the following classifiers for the classification of the given dataset. The classifiers namely are:
    - Decision Tree
    - Multilayer Perceptron
    - Logistic Regression
    - Kernel SVM

- Then we have applied all the four classifiers to the following form of dataset:
    - Raw Data
    - After applying PCA to the dataset.
    - After applying LDA to the dataset.

- One more interesting point to note is that we have also tried and changed the number of dimensions while applying the LDA as well as PCA.

- Here is a list of parameters that are being varied while the application of the classifier on the dataset.
    - Decision Tree

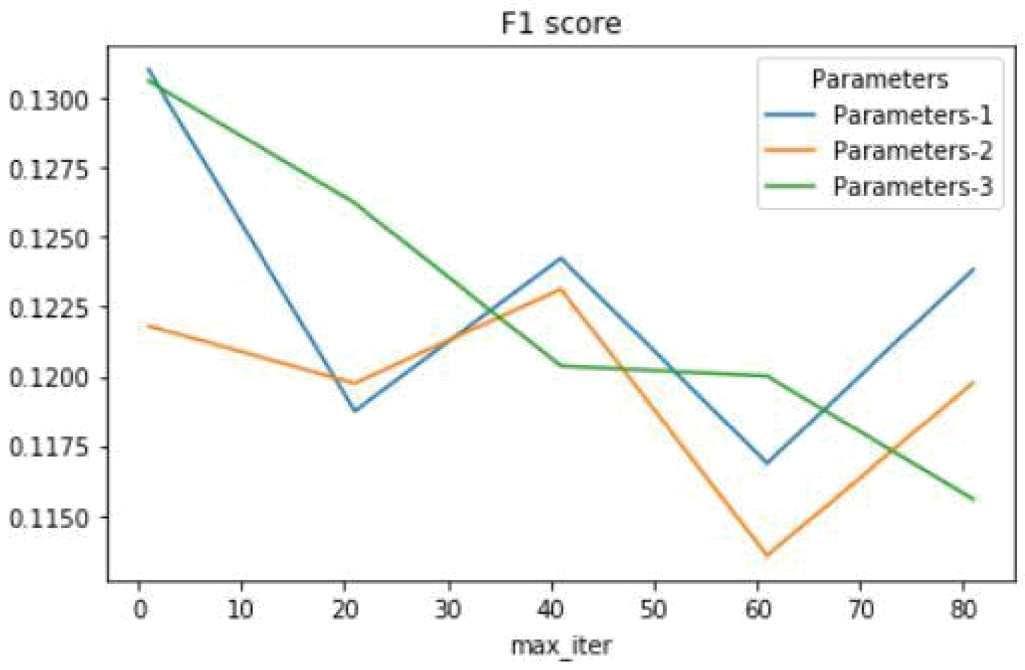
        - Max_depth = This parameter of the decision tree classifier specifies the depth till which the classifier works.
        - criterion : This specifies the criterion of the classification.
    - Logistic Regression
        - C: The threshold after which the iteration breaks
        - Max_iterations: This defines the maximum number of iterations while converging to the weights.
    - Kernel SVM
        - Max_iterations
        - C
    - MLP
        - Alpha
        - Hidden Layer Size
    - PCA
        - Number of dimensions
    - LDA
        - Number of Dimensions

RESULTS

1.1) Logistic Regression using PCA

This is the table of the results by applying the pca to the dataset and then applying logistic regression for classification. We vary the number of components while applying the pca and then vary the threshold while using the logistic regression. The table is shown below

SR NO.	n_ components	C	train_score	test_score
1	8	1	.306	.2901
2	16	1	.3295	.3385
3	32	1	.3834	.3599
4	64	1	.4032	.3688
5	8	0.05	.3092	.2932

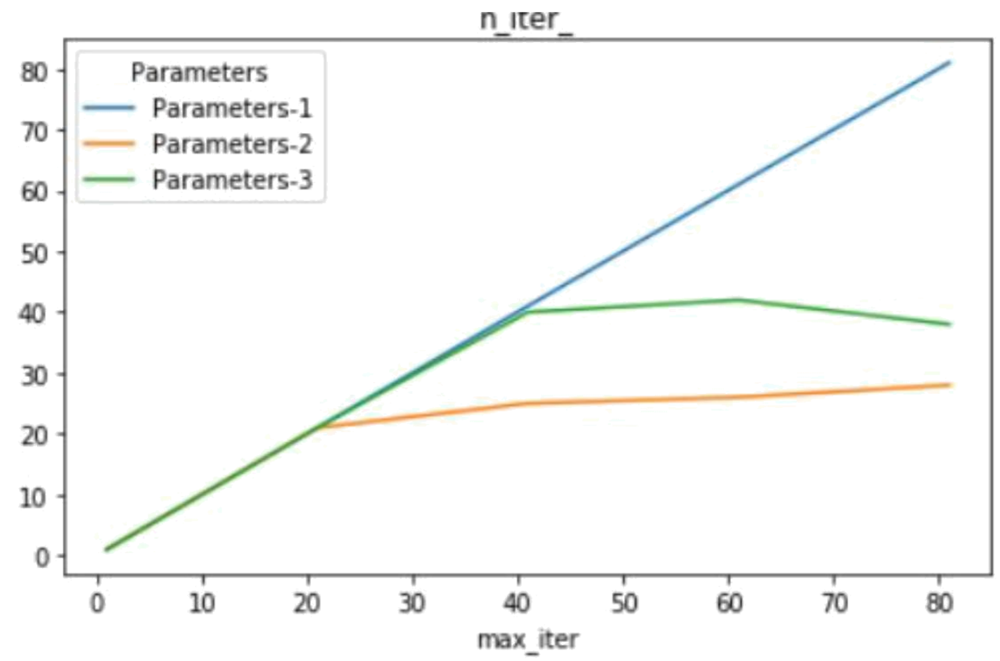


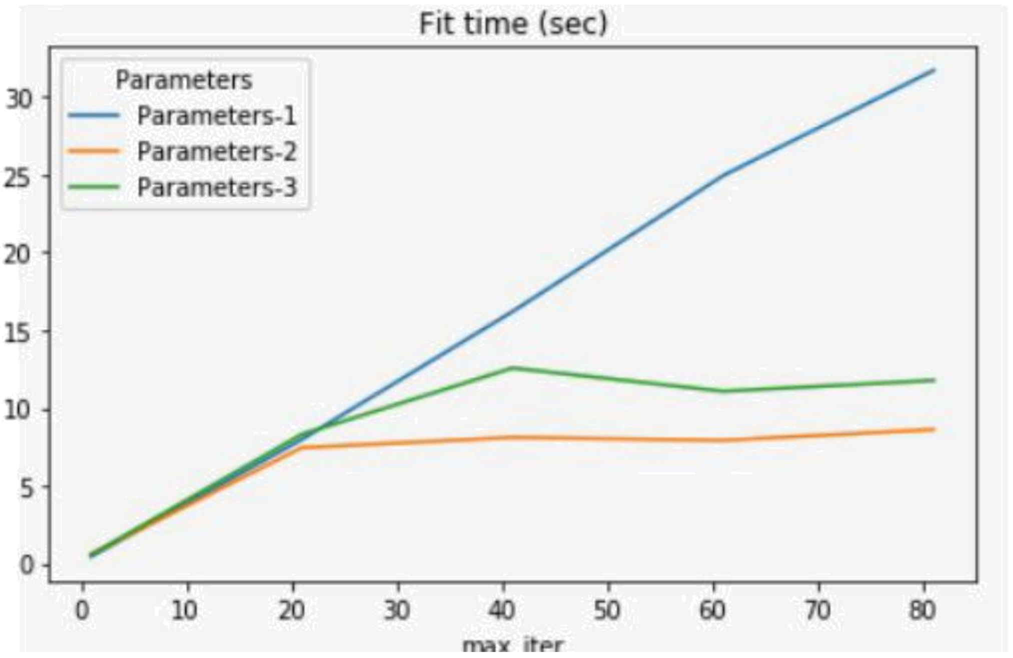
1.2) **Logistic regression on raw data**

In this we can clearly observe that the score increases on increasing the number of iteration. The maximum iterations for the below drawn table is taken to be 1000 iterations.

SR NO.	C	Train_score	Test_Score
1	1	.29293	.28343
2	.05	.30102	.28283
3	.001	.28680	.28243

Here are the graphs for the same,





1.3) **Logistic regression using LDA**

In this we can clearly see that the the training as well as the testing score increases on increasing the number of dimensions and decreasing the value of C in logistic Regression.

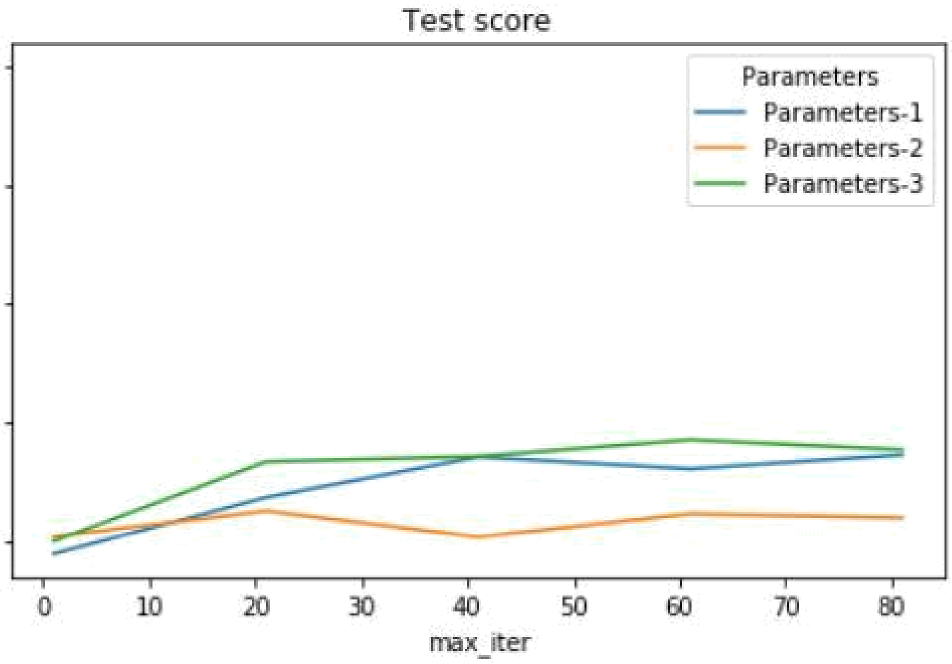
Sr no.	C	Dimensions	Train Score	Test Score
1	1	3	.24002	.23927
2	1	6	.27987	.28125
3	.05	3	.23787	.23775
4	.05	9	.31868	.30687

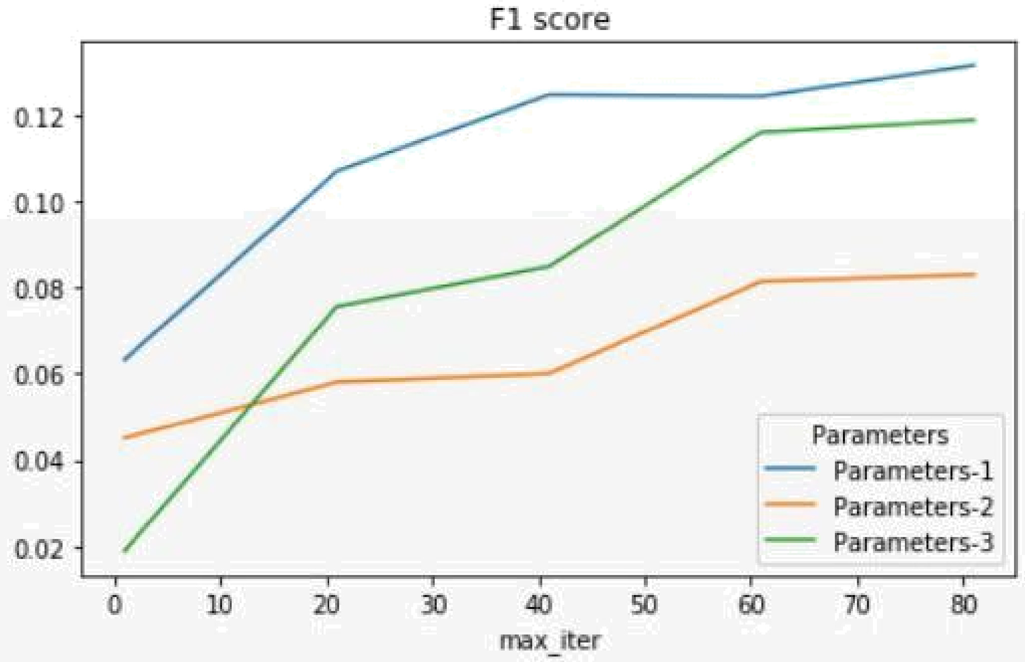
Hyperparameter Tuning:In this we vary 2 parameters, C and the number of dimensions in the lda. As we know that the C is the inverse of the regularization strength, that is the smaller the value of C the larger will be the strength of the regularisation. Also we can conclude that the more the number of dimensions the more will be the accuracy as the amount of information retained is also more.

2.1) Using Kernel SVM along with the PCA.

We have here varied the number of components in PCA and the values of C and gamma in the kernel SVM.

	lda__n_components	svm__C	svm__gamma	mean_train_score	mean_test_score
0	3	1	0.0001	0.518081	0.169368
1	3	1	1E-06	0.103	0.103
2	3	1	1E-05	0.103	0.103
3	3	1	1E-07	0.103	0.103
4	3	0.05	0.0001	0.103	0.103
5	3	0.05	1E-06	0.103	0.103
6	3	0.05	1E-05	0.103	0.103
7	3	0.05	1E-07	0.103	0.103
8	3	0.001	0.0001	0.103	0.103
9	3	0.001	1E-06	0.103	0.103
10	3	0.001	1E-05	0.103	0.103
11	3	0.001	1E-07	0.103	0.103
12	6	1	0.0001	0.686752	0.186622
13	6	1	1E-06	0.103	0.103
14	6	1	1E-05	0.103	0.10325
15	6	1	1E-07	0.103	0.103
16	6	0.05	0.0001	0.103	0.103
17	6	0.05	1E-06	0.103	0.103
18	6	0.05	1E-05	0.103	0.103
19	6	0.05	1E-07	0.103	0.103
20	6	0.001	0.0001	0.103	0.103
21	6	0.001	1E-06	0.103	0.103

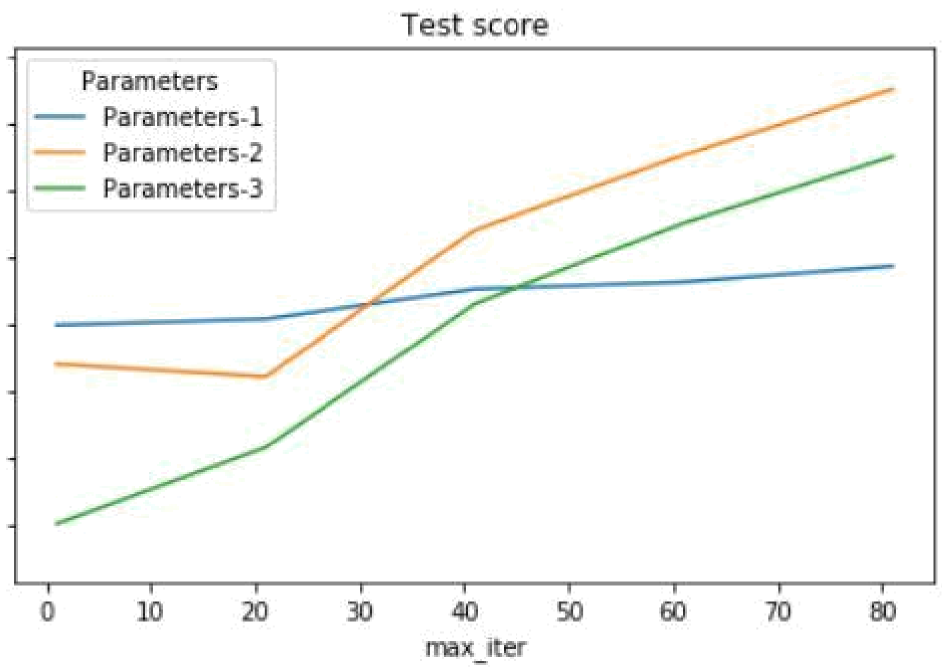

Test score
Parameters
Parameters-1
Parameters-2
Parameters-3
0
10
20
30
40
50
60
70
80
max_iter

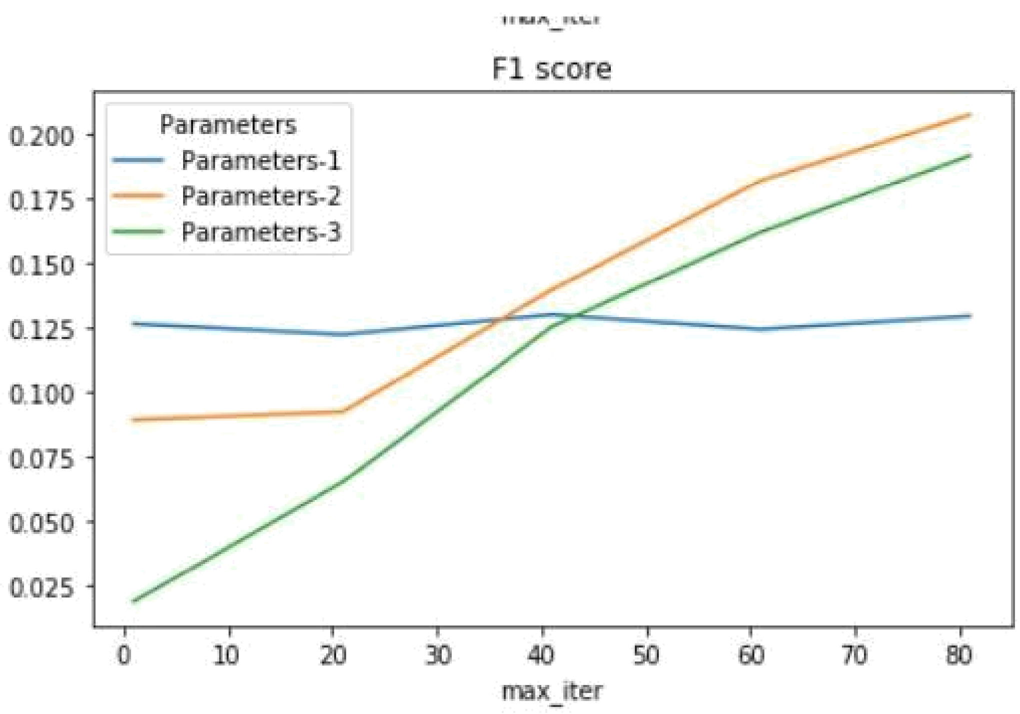

F1 score
0.12
0.10
0.08
0.06
0.04
0.02
Parameters
Parameters-1
Parameters-2
Parameters-3
0
10
20
30
40
50
60
70
80
max_iter

2.2) Kernel SVM along with Raw data

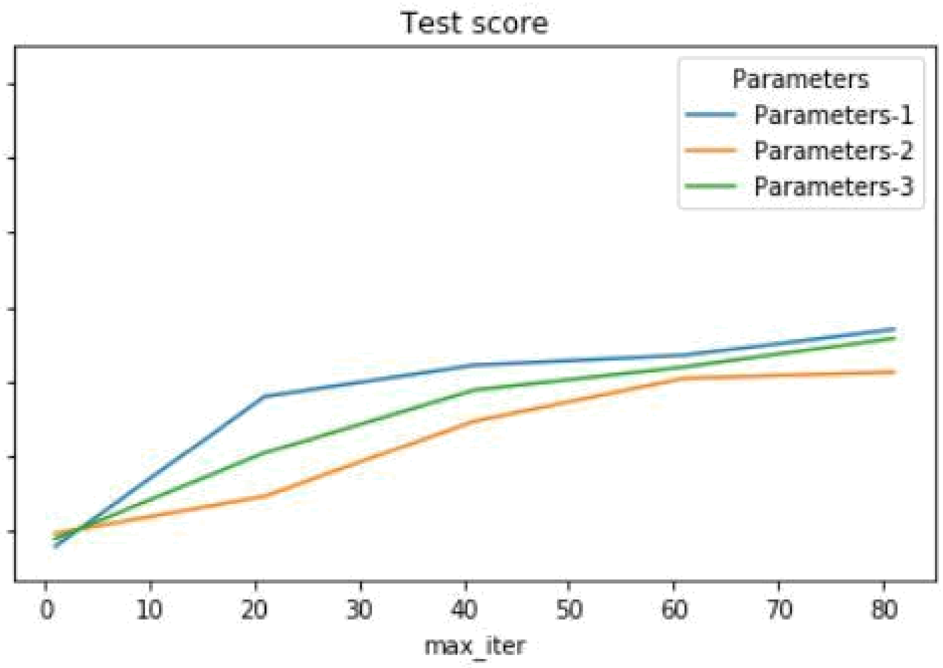
We can see that the using kernel SVM along with raw data, it takes a lot of computational time and thus it is quite complex to compute without doing dimension reduction. Here also we have varied the values of gamma as well as the values of C in SVM.

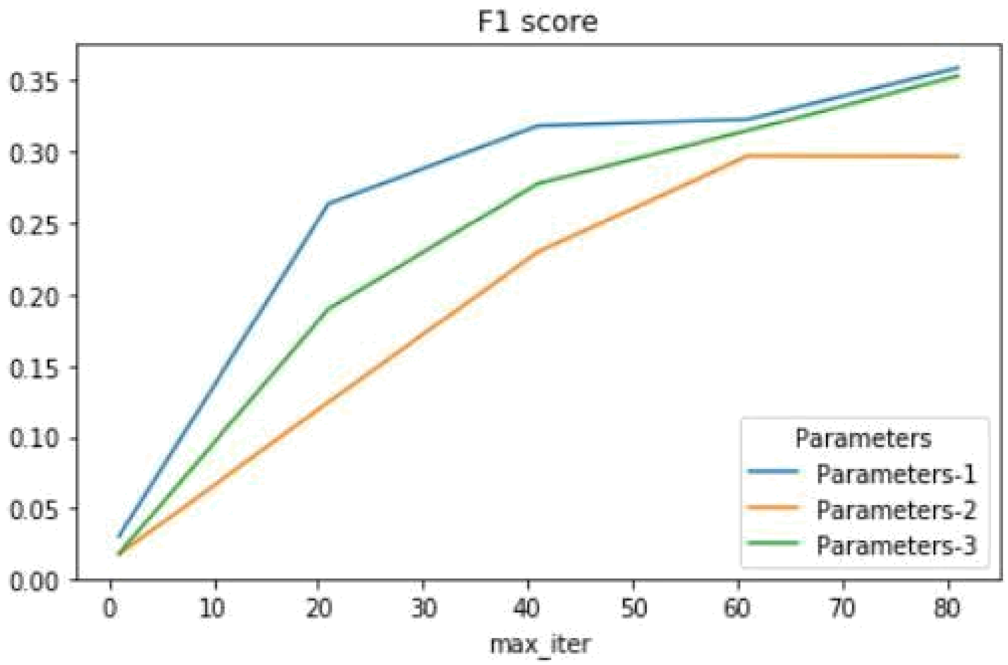
Sr no.	C	Gamma	Train score	Test score
1	1	1e-3	.96	.1067
2	1	1e-6	.7687	.3223
3	1	1e-5	.9842	.1337
4	.05	1e-3	.1048	.1058
5	.05	1e-6	.1668	.1563
6	.05	1e-5	.1048	.0962



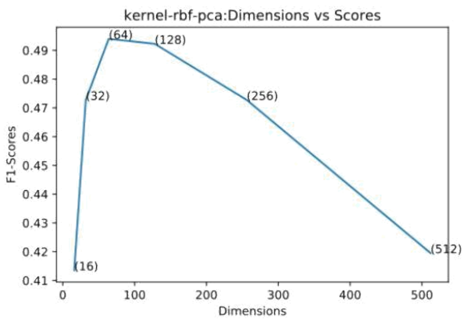


2.3) Kernel SVM along with LDA applied to the training and testing data

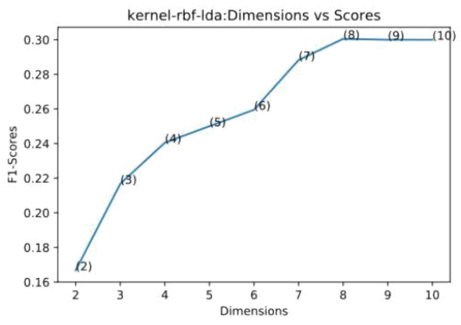




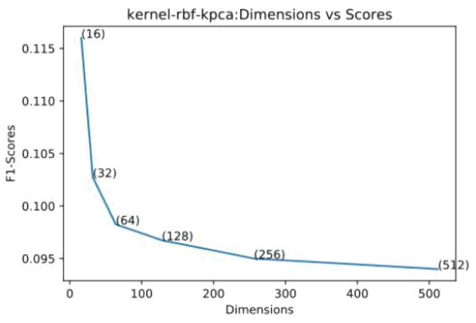
Graphs for the F1 score for all the three against the number of dimensions.



(a) PCA



(b) LDA

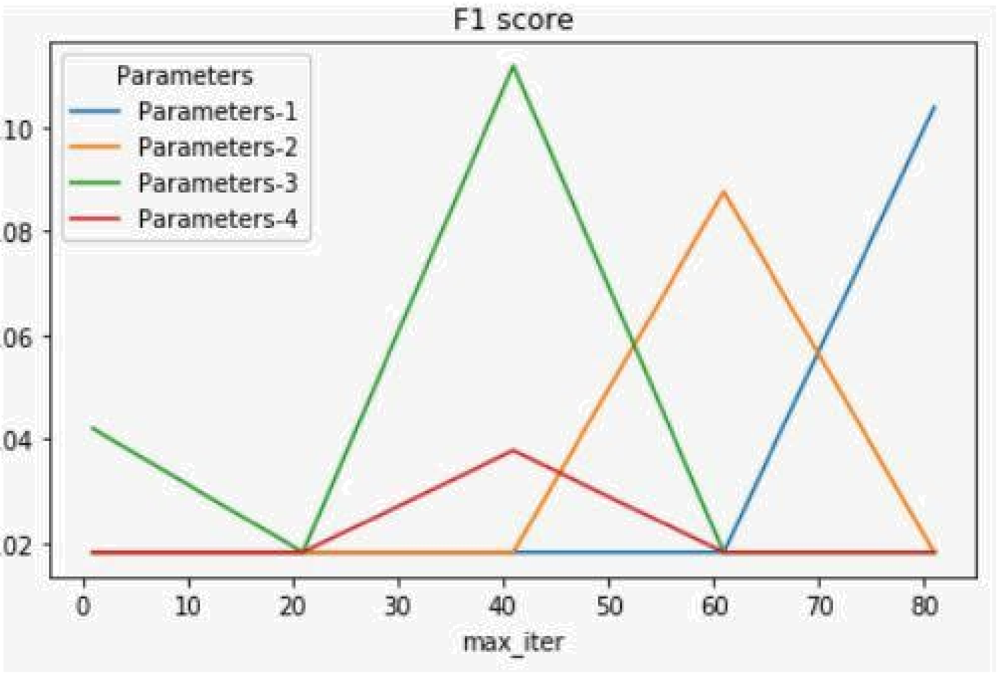


Hyperparameter tuning:

As we are varying the number of dimensions in the PCa and LDA as well as the value of C and gamma in the Kernel SVM we can see that on increasing the number of dimensions we are getting higher accuracy and vice versa.
Where C is the penalty parameter for the error term and the gamma is the kernel coefficient. For this we can see that the best score in PCA is observed in the case of lowest gamma and largest C. Where as in LDA best is observed in the largest C and largest gamma.

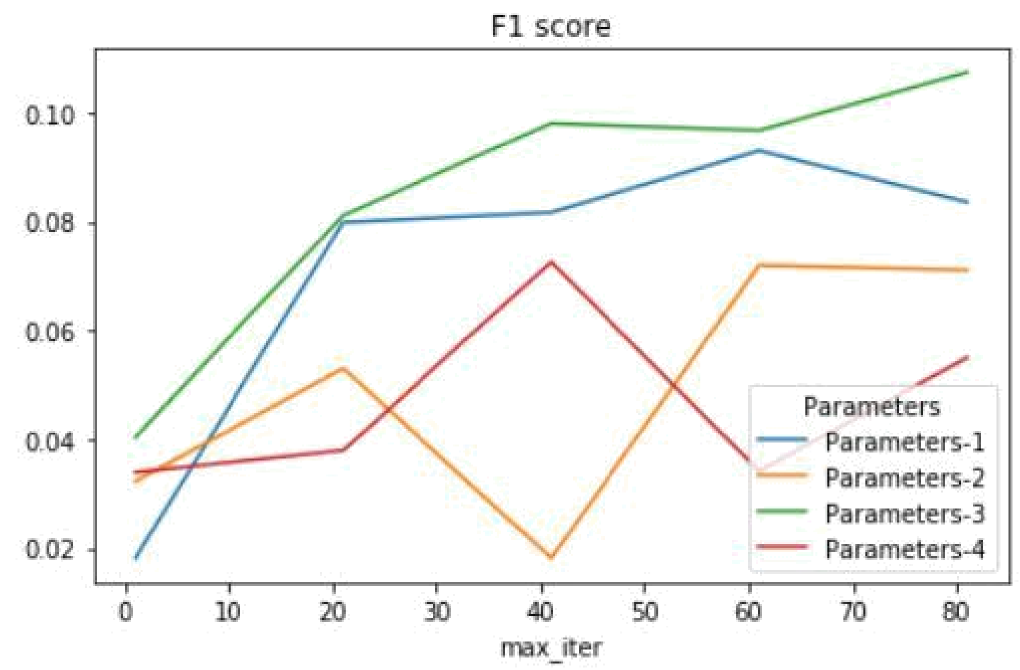
3.1)MLP using raw data

Sr no.	max_iter	F1 score	Test score
1	10	.0189	.0918
2	20	.01812	.10
3	40	.01765	.10076
4	60	.103857	0.1870



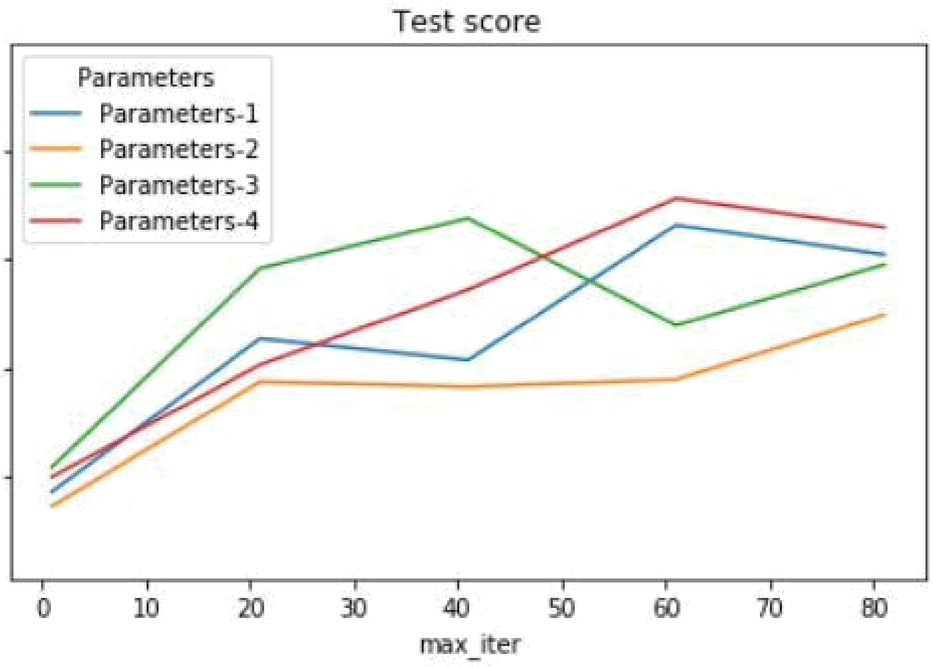
3.2)MLP using PCA to the training and the testing dataset.

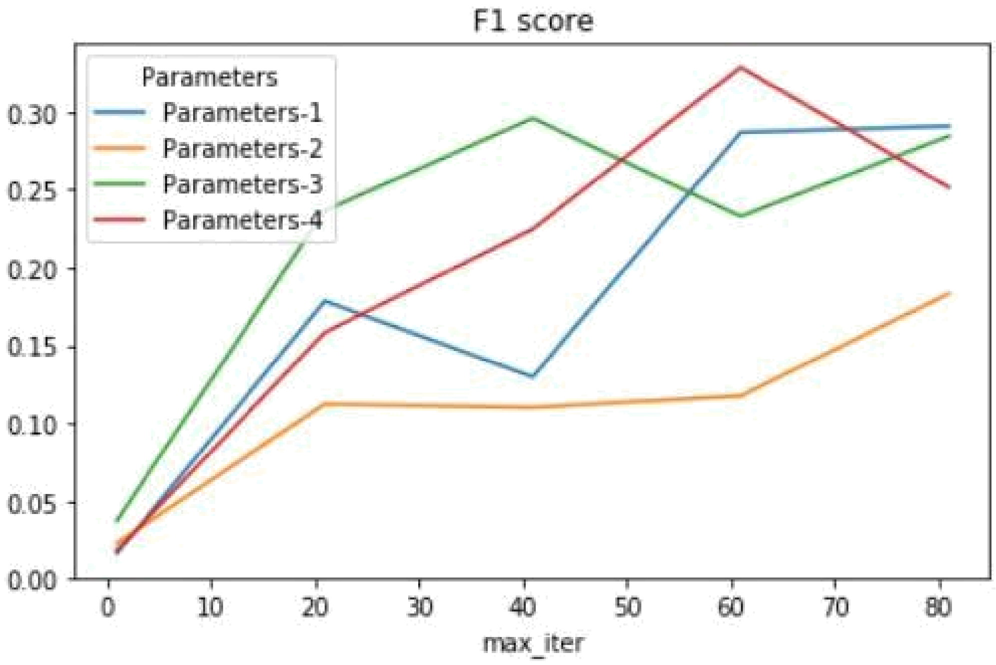
3.3) MLP using the LDA



Sr no.	iterations	dimensions	F1 score	Test score
1	10	8	.1826	.1382
2	20	16	.2351	.2478
3	30	32	.2877	.2619
4	40	64	.3035	.3009
5	50	32	.3871	.3921

3.3) MLP using the LDA





Sr no.	iterations	dimensions	Test score	F1 score
1	10	32	.3482	.3309
2	30	32	.4692	.4497

3	60	32	.5001	.5118

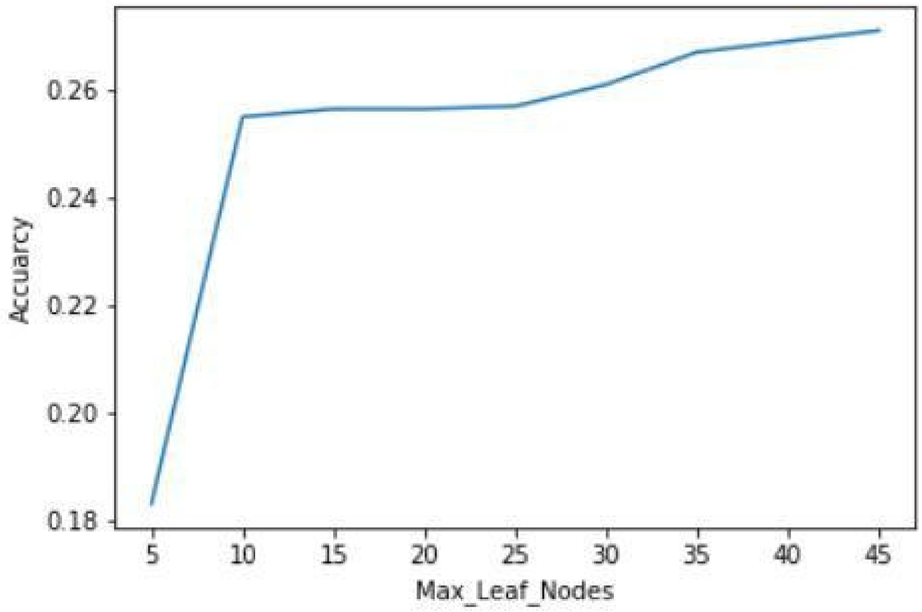
4.1)Decision Tree on the raw data

In the decision tree classifier what we have done is we have varied the depth of the tree till which classification happens. The graph for the same is shown below

4.2)Decision Tree along with the PCA train set and test set

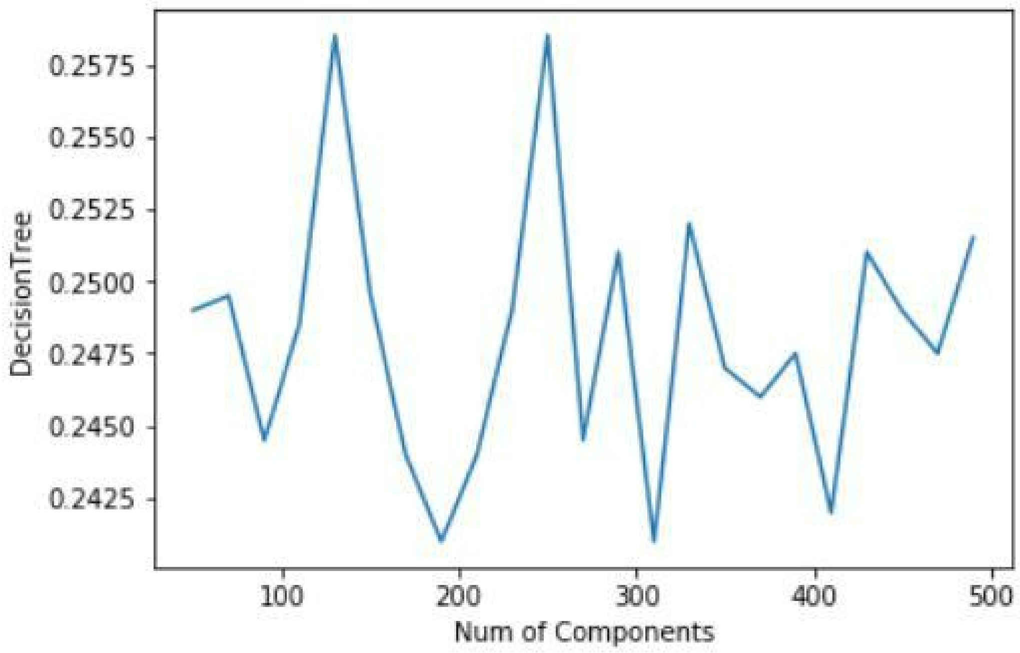
The above part shows the variation of the accuracy on varying the depth of the decision tree and the below graph shows the variation of the accuracy on varying the number of principal components in to the consideration. We can see how fluction it shows on increasing the number of dimensions.

Accuracy plotted against the varying the value of the maximum number of leaf nodes required for the growth of the tree.

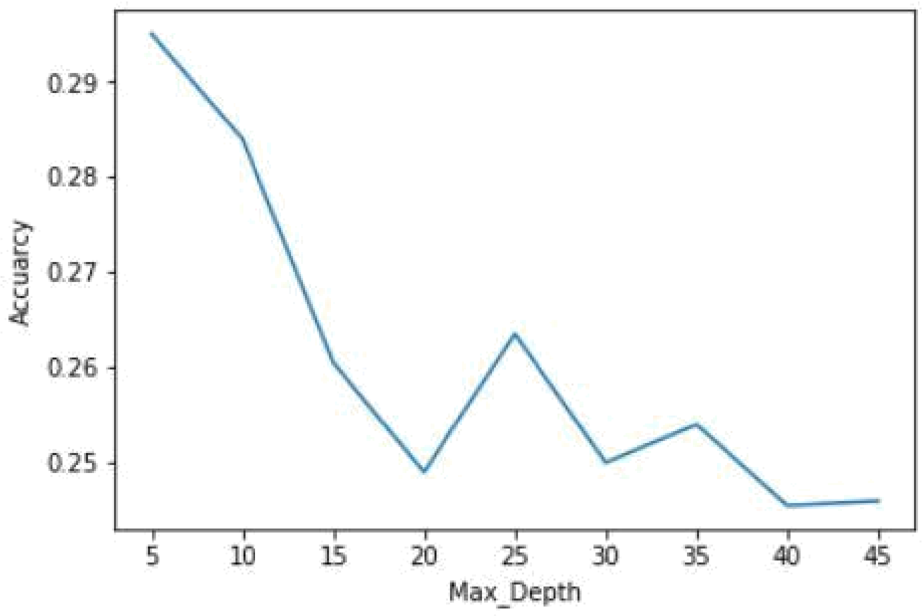


4.3) Decision Tree along with LDA

The graph plotted of accuracy vs the number of dimensions in the LDA dimension reduction.



The graph plotted for the varying the maximum depth of the tree is shown below:



THE PROBLEM OF OVERFITTING:

We all know that overfitting is a common phenomenon that can occur while classification. For observing the same we have plotted the training as well as the testing dataset together. The training data is in red where as the testing data is in blue. The following observations are made from the plots.

- The test data accuracy reaches a peak and then starts decreasing on further increasing the number of components.But, in theory, the accuracy should increase on increasing the number of components due to increase in the number of information with less data compression.

- The variation here is seen due to overfitting of the training set, which leads to more misclassification of the test data. As we tweak the hyperparameter to get more and more accuracy on the training set, we overfit it and thus misclassification increases on the test set.

- In the decision tree classification, on increasing the number of the leaf nodes or on increasing the depth, thus we are trying to focus on the increasing the number of features and thus trying to fit the training set. This is example of overfitting in the decision tree classification.