SMAI -MINI PROJECT 2

Implementing Data Classifiers for CIFAR-10 dataset

SathyaSravya.V
20161121

For the given CIFAR-10 data,loading is done
Later,the train data is split in the ratio 80:20 for using as training and testing sets,using train_test_split ,putting test_size =0.2.
This is followed by preprocessing (function preprocessing is called from main,results are stored in trprep_data and teprep_data),
dimensionality reduction functions corresponding to PCA and LDA are called and results are stored in te_data1,tr_data1 and tr_data2,te_data2 in the main function.

So the three data presentations taken are

1) preprocessed data(normalised and centered)(RAW DATA)
2) PCA
3)LDA

Classifiers taken:
1) CART/Decision Tree
2) MLP
3) Kernel SVM with RBF Kernel
4) Logistic Regression

Main observations

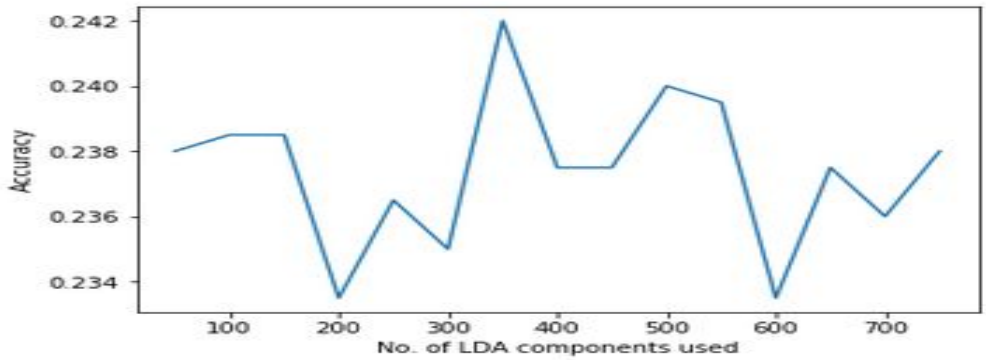
CART/Decision Tree:

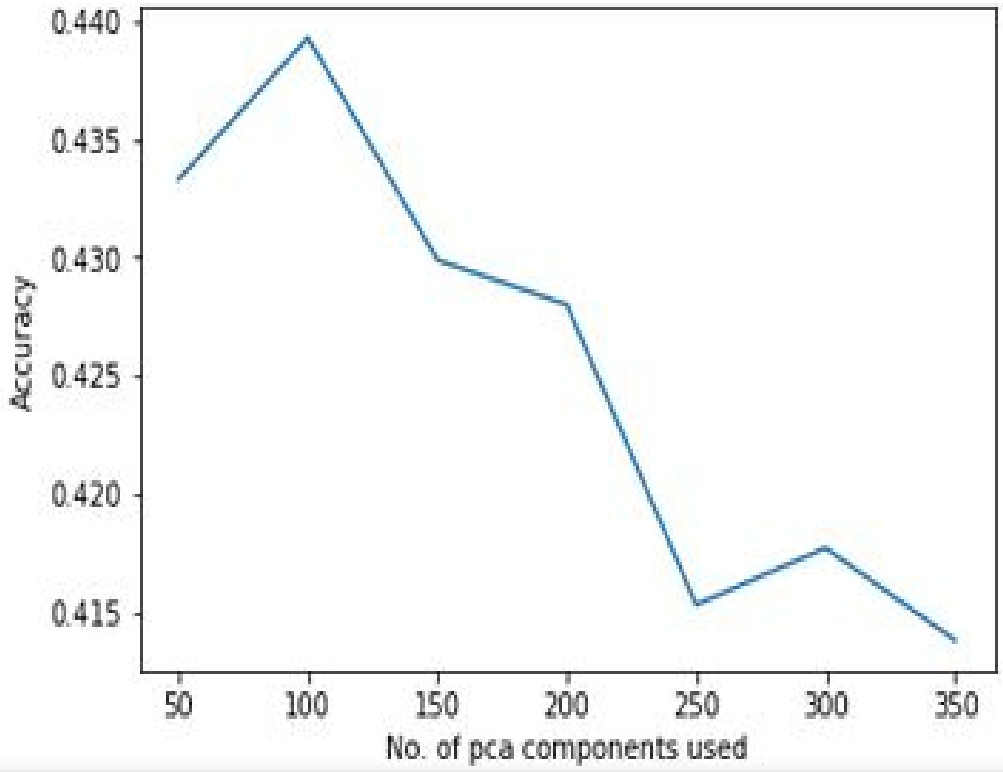
Hyperparameters taken into consideration are,

- n_estimators i.e number of trees
- min_samples_split
- max_depth
- Number of pca components required
- Number of lda components required

Accordingly functions vary_num_trees , vary_min_split_tree ,vary_depth_tree , vary_num_pca ,vary_num_lda are written varying those respective parameters and results are plotted on graphs.
For 100 pca components taken,maximum accuracy is seen. And LDA components can reach upto 350.

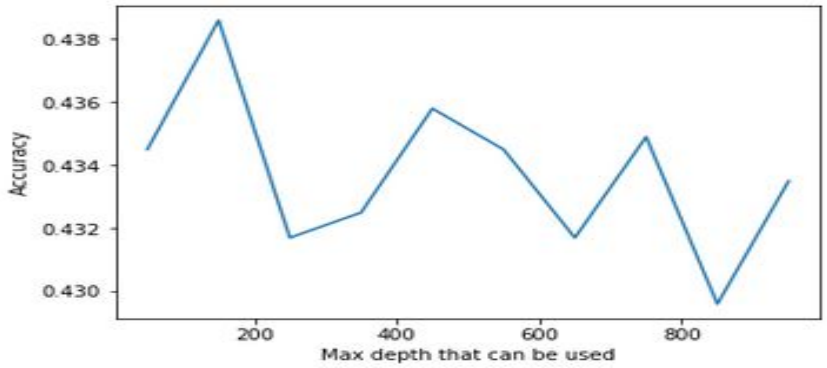
val - F1 score: 0.238
Accuracy: 0.238





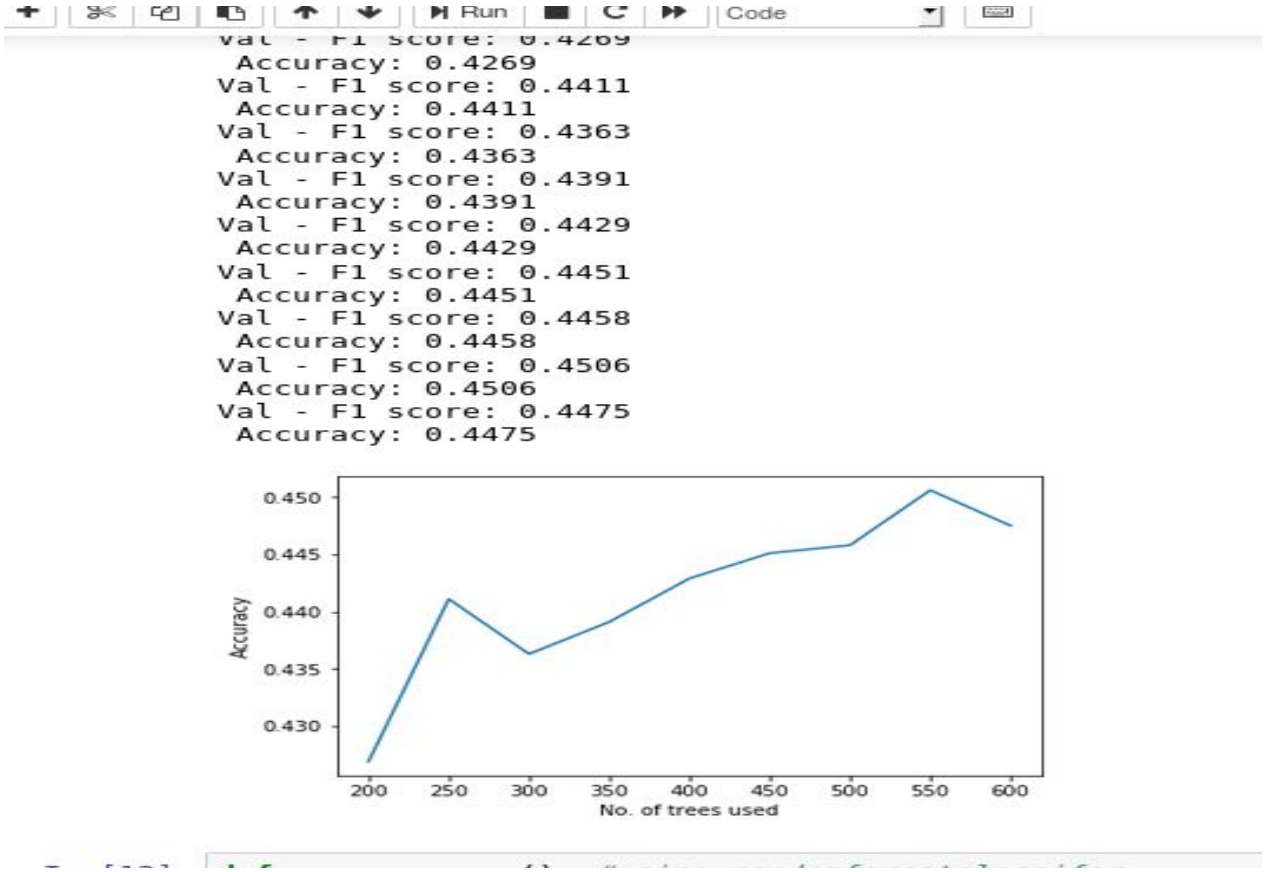
- For 300 trees,5 splits taken the maximum depth varied like

```
 Accuracy: 0.4345
Val - F1 score: 0.43859999999999993
 Accuracy: 0.4386
Val - F1 score: 0.4317
 Accuracy: 0.4317
Val - F1 score: 0.4325
 Accuracy: 0.4325
Val - F1 score: 0.4358000000000001
 Accuracy: 0.4358
Val - F1 score: 0.4345
 Accuracy: 0.4345
Val - F1 score: 0.4317
 Accuracy: 0.4317
Val - F1 score: 0.43489999999999995
 Accuracy: 0.4349
Val - F1 score: 0.42960000000000004
 Accuracy: 0.4296
Val - F1 score: 0.4335
 Accuracy: 0.4335
```



Telling that max accuracy will be obtained at maximum depth greater than (number of trees /2). (here max_depth = 180 nearly).

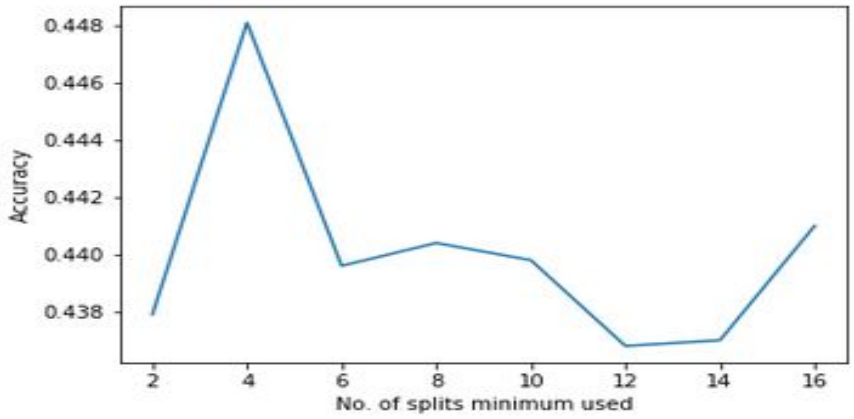
Supporting the same , when depth is 450 ,splits =5 , number of trees varied like ,



Showing maximum accuracy at 570 trees.

For 570 trees taken and 455 max_depth taken , I got the number of minimum splits variation with accuracy like shown below,reporting max acc. At 4 splits.

```
100%|██████████| 1/1 [00:00<00:00, 24.47it/s]
Val - F1 score: 0.4379
 Accuracy: 0.4379
Val - F1 score: 0.44810000000000005
 Accuracy: 0.4481
Val - F1 score: 0.4396
 Accuracy: 0.4396
Val - F1 score: 0.4404
 Accuracy: 0.4404
Val - F1 score: 0.4398
 Accuracy: 0.4398
Val - F1 score: 0.4368000000000001
 Accuracy: 0.4368
Val - F1 score: 0.437
 Accuracy: 0.437
Val - F1 score: 0.441
 Accuracy: 0.441
```



Classifier	Features	Accuracy	F1-score
CART/Decision Tree	raw-pixels	43.23%	0.4323
CART/Decision Tree	principle components	44.48%	0.4448
CART/Decision Tree	LDA reduction	24.34%	0.2434
Kernel SVM with RBF Kernel	raw-pixels	47.05%	0.4705
Kernel SVM with RBF Kernel	principle components	47.05%	0.4705
Kernel SVM with RBF Kernel	LDA reduction	26%	0.2535
MLP	raw-pixels	41.16%	0.4116
MLP	principle components	47.14%	0.4714
MLP	LDA reduction	34.94%	0.3493
LR	raw-pixels	26.8%	0.268
LR	principle components	37.4%	0.374
LR	LDA reduction	22.95%	0.2295

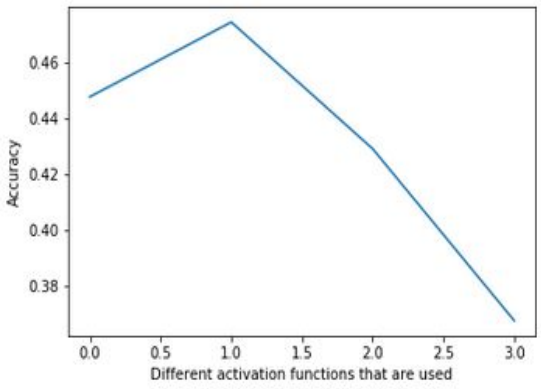
All above listed are relatively good results.They reveal that ,
MLP > Kernel SVM with RBF Kernel > CART/Decision tree >LR for raw components .
Also MLP was the fastest classifier obtained so far,took less time to produce results.

MLP:

Changing the following hyperparameters

- Activation function
- Number of hidden layers
- Neurons within a hidden layer

```
100%|██████████| 1/1 [00:00<00:00, 13.20it/s]
Test set score: 0.447500
Val - F1 score: 0.4475
 Accuracy: 0.4475
Test set score: 0.474200
Val - F1 score: 0.4742
 Accuracy: 0.4742
Test set score: 0.428900
Val - F1 score: 0.4289
 Accuracy: 0.4289
Test set score: 0.367500
Val - F1 score: 0.3675
 Accuracy: 0.3675
```

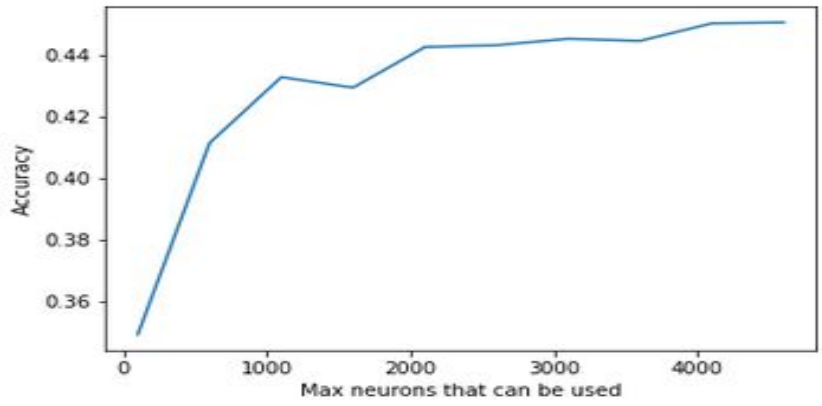


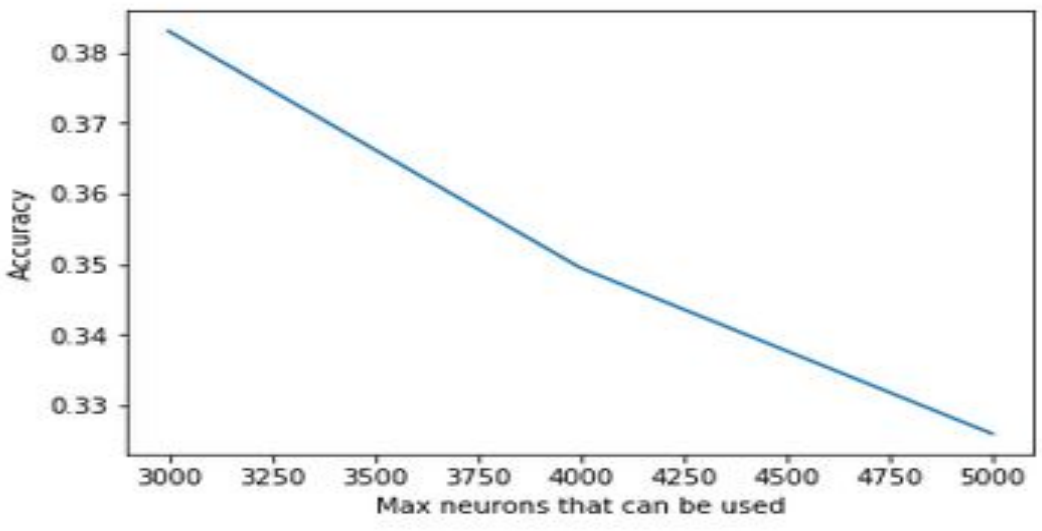
When activation function used are varied, I got maximum accuracies in the order ,

Relu > logistic > tanh > identity .

- When I varied the number of neurons within a layer , the accuracy increased with number of neurons when there is only 1 layer like shown below.
- Also for a given number of neurons(n1) in first layer,if we put one more layer the accuracy increased till (n2)number of neurons where n2 is not far greater than n1.

```
 Accuracy: 0.4113
Test set score: 0.432700
Val - F1 score: 0.4326999999999999
 Accuracy: 0.4327
Test set score: 0.429300
Val - F1 score: 0.4293
 Accuracy: 0.4293
Test set score: 0.442500
Val - F1 score: 0.4425
 Accuracy: 0.4425
Test set score: 0.443100
Val - F1 score: 0.4431
 Accuracy: 0.4431
Test set score: 0.445200
Val - F1 score: 0.4452
 Accuracy: 0.4452
Test set score: 0.444500
Val - F1 score: 0.4445
 Accuracy: 0.4445
Test set score: 0.450200
Val - F1 score: 0.4502
 Accuracy: 0.4502
Test set score: 0.450500
Val - F1 score: 0.4505
 Accuracy: 0.4505
```





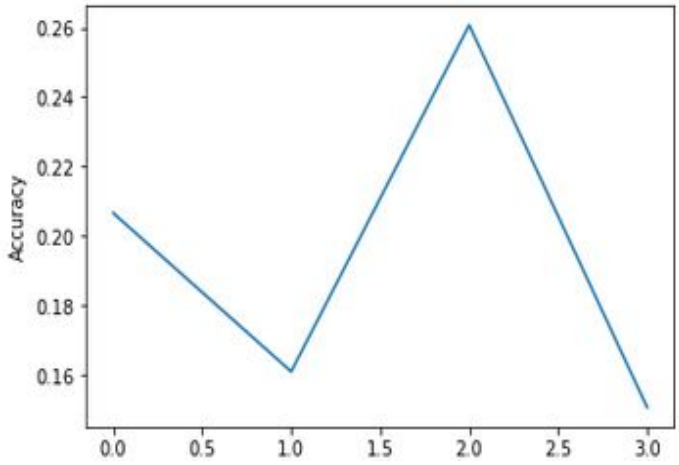
- Also using more than 2 hidden layers resulted in a great drop in accuracy .That is why I limited changing parameters of 1 and 2 layers.
-

No.of hidden layers (hidden_layer_size)	Accuracies
(500,)	46%
(500,500,)	47.5%
(500,500,500,)	45.8%
(500,500,500,500,)	45%
(500,500,500,500,500,)	44%

-

SVM :

- **Kernels**
- **Max iterations**
- **Number of pca components**

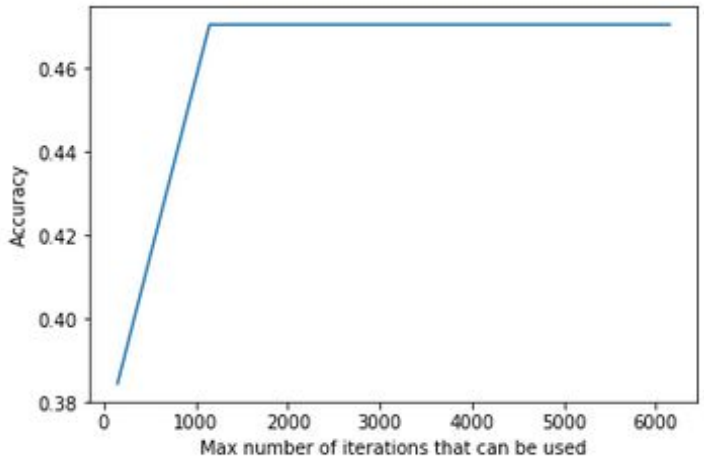


By varying kernels ,accuracies are determined to vary like shown below

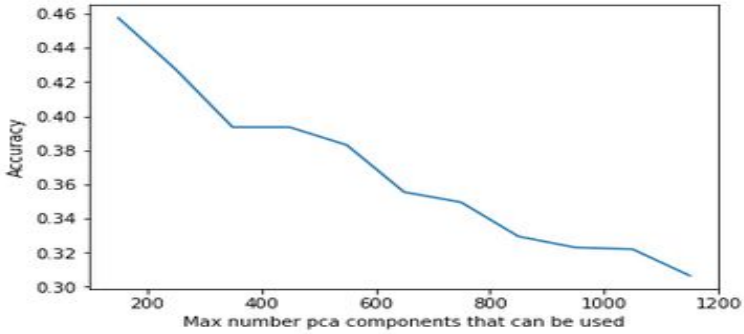
rbf > linear > polynomial > sigmoid

- Reducing tolerance decreased the accuracy and f1 score.
- Better accuracy was obtained at less number of iterations itself,later it doesn't change.(at 1500 iterations best accuracy of 47% is obtained)

```
Val-PCA reduced data -SVM - F1 score: 0.4705
 Accuracy: 0.4705
Val-PCA reduced data -SVM - F1 score: 0.4705
 Accuracy: 0.4705
```



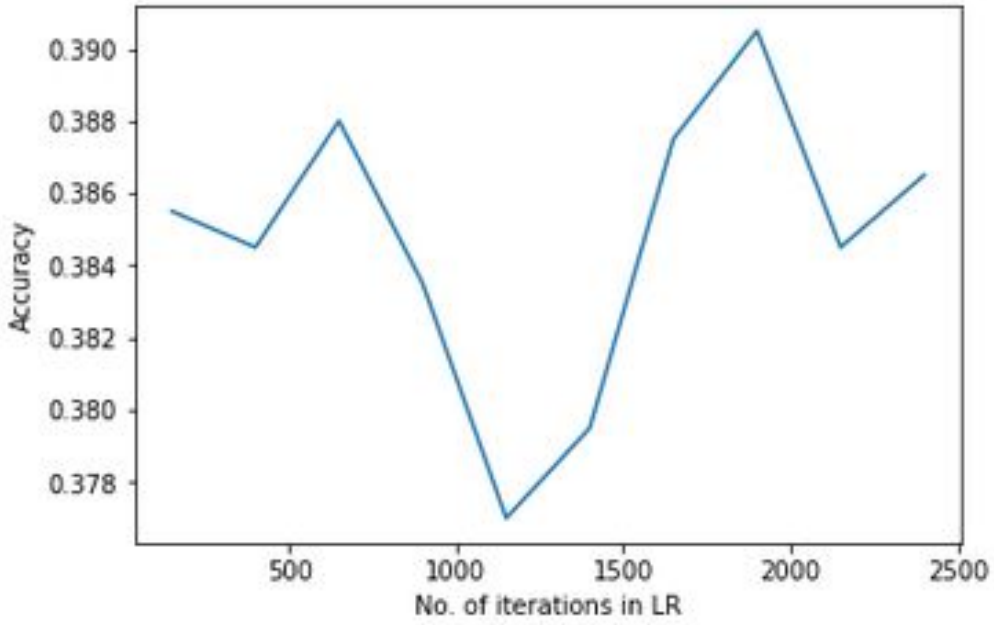
```
 Accuracy: 0.4275
Val-PCA reduced data -SVM - F1 score: 0.3935
 Accuracy: 0.3935
Val-PCA reduced data -SVM - F1 score: 0.3935
 Accuracy: 0.3935
Val-PCA reduced data -SVM - F1 score: 0.383
 Accuracy: 0.383
Val-PCA reduced data -SVM - F1 score: 0.35550000000000004
 Accuracy: 0.3555
Val-PCA reduced data -SVM - F1 score: 0.3495
 Accuracy: 0.3495
Val-PCA reduced data -SVM - F1 score: 0.3295
 Accuracy: 0.3295
Val-PCA reduced data -SVM - F1 score: 0.323
 Accuracy: 0.323
Val-PCA reduced data -SVM - F1 score: 0.322
 Accuracy: 0.322
Val-PCA reduced data -SVM - F1 score: 0.3065
 Accuracy: 0.3065
```



LR (LOGISTIC REGRESSION):

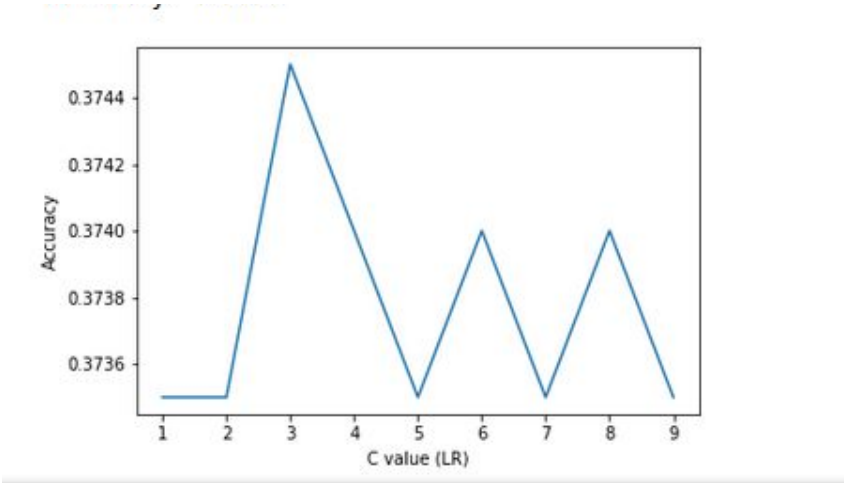
- **Number of iterations**
- **C factor**
- **Number of pca components**

```
 Accuracy: 0.3845
Val - F1 score: 0.38649999999999995
 Accuracy: 0.3865
```

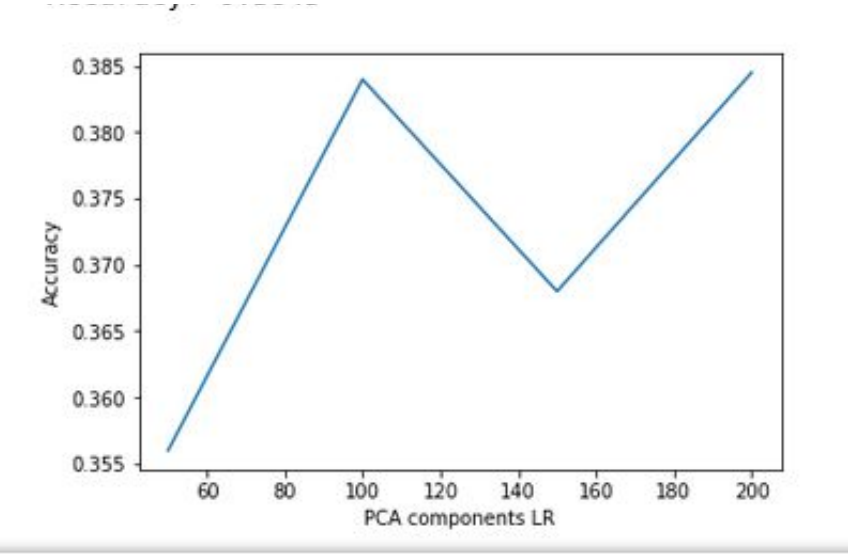


The number of iterations for which maximum was obtained is around 1800.

- Limiting this would reduce accuracy by minimum amount.
- Also the value of ‘C’ for which best accuracy came is around 3.



- Reducing tolerance decreased the accuracy and f1 score.
- Also number of LDA components required are comparably more than number of pca components to attain same accuracy .



Thus best possible accuracies are obtained for above 4 classifiers selected ,after good amount of playing with hyperparameters and different parameters are understood from above results.

On the whole , accuracies would be like ,

Classifier /data type	**Raw data(around)**	**Pca**	**Lda**
Decision tree	43% (42-44)	44%(43-45)	24%(22-26)
Svm (rbf)	47%(46-48)	47.05%(46-48)	25%(24-27)
MLP	42%(41-44)	47.14%-48%	34% (30-35)
LR	27%(26-30)	37-39%	22%(21-25)

Using raw data without processing leads to good accuracies and good classifiers development during training but it takes lot of time .

Problem of overfitting and thoughts (solution): This problem's severity is measured by the difference in the accuracies of training set and test sets of data.

- Also the decrease in accuracy for increase in number of iterations for LR and SVM reveal this overfitting.
- The increase in number of neurons used beyond 3000 in MLP reveal this problem(since accuracy decreases)
- The increase in number of pca ,lda components beyond 100,400,depth of tree and number of trees beyond respective limits too reflect this problem.

This problem though occurred can be solved by reducing the strength of classifiers above,as in selecting parameters such that the accuracy is moderate and not very high.This helps in building a relatively weak classifier thereby reducing the problem of overfitting to an extent.Since we already obtained variation of different parameters like 'C' , number of components during reduction ,depth of tree,number of splits etc we can decrease them and observe difference.We can decrease tolerance in SVM , LR,CART methods.And decreasing number of hidden layers in MLP method.

Also training on two different datasets help in solving this(distributions of data with good variance).Like using different batches of data at a time to train in given cifar dataset would help.But this in turn requires more training time and it's power consuming.

Practical problems:

- Also the training time ,testing time is very long for all above analysis done.Playing with hyperparameters had been difficult because ,repetitive execution of code required large amounts of time and power.

- Also I have tested and obtained best values of hyper parameters individually,which resulted in getting best of their values when others are kept constant.
- All best values of parameters obtained ,may not result in best accuracy when applied.They may not work together in groups because of dependencies they may have like in the case of decision trees , min_split in tree ,max_depth ,num_of_trees depend on each other possibly.
- It would be better than this if dependencies like these are understood better and trained .