PRNN ASSIGNMENT - 3

p1: Decision Tree Classifier

In this report, we implemented classification trees using Gini impurity and entropy as impurity functions, with different depths, for the MNIST problem using the PCA data.
First, we implemented the classification tree using Gini impurity as the impurity function. We trained the tree on the training set and evaluated its accuracy on the test set for different maximum depths. The results are shown below:

For p3 dataset in A1: (Gini Impurity)

For depth 5 and minimum samples per split allowed at each node is 3.(This can also act as regularizer).

Classification Accuracy on Test Data is: 51.19

Confusion Matrix is:

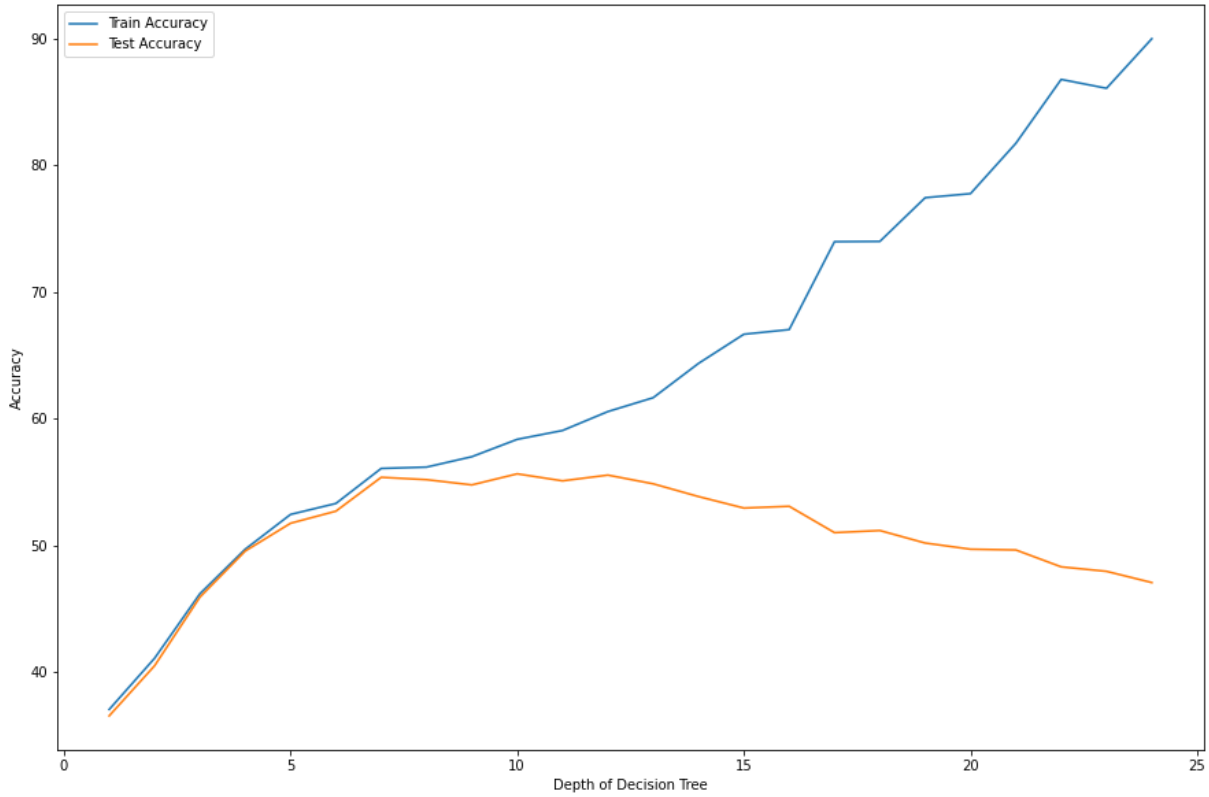
[[2203. 185. 139. 81. 363.]
[1237. 1255. 134. 50. 305.]
[1116. 147. 1441. 49. 287.]
[1331. 167. 106. 1089. 282.]
[1027. 161. 90. 64. 1691.]]

f1 Score of Class 1 is: 0.4457258472432979
f1 Score of Class 2 is: 0.5126633986928105
f1 Score of Class 3 is: 0.5822222222222222
f1 Score of Class 4 is: 0.5055710306406686
f1 Score of Class 5 is: 0.5673544707263881

For different depths:(Bias-Variance)

Region	Depth	Train Accuracy	Test Accuracy
Underfitting	1	37.065	36.56
Underfitting	3	46.192	45.92
	4	49.707	49.56
	6	53.315	52.71
	7	56.08	55.38
	8	56.185	55.19
	11	59.073	55.1
Overfitting	17	73.973	51.01
Overfitting	20	77.763	49.71
Overfitting	24	89.988	47.07

Bias-Variance For Gini Impurity, Accuracy vs Depth for Gini Impurity(For p3 dataset in A1)

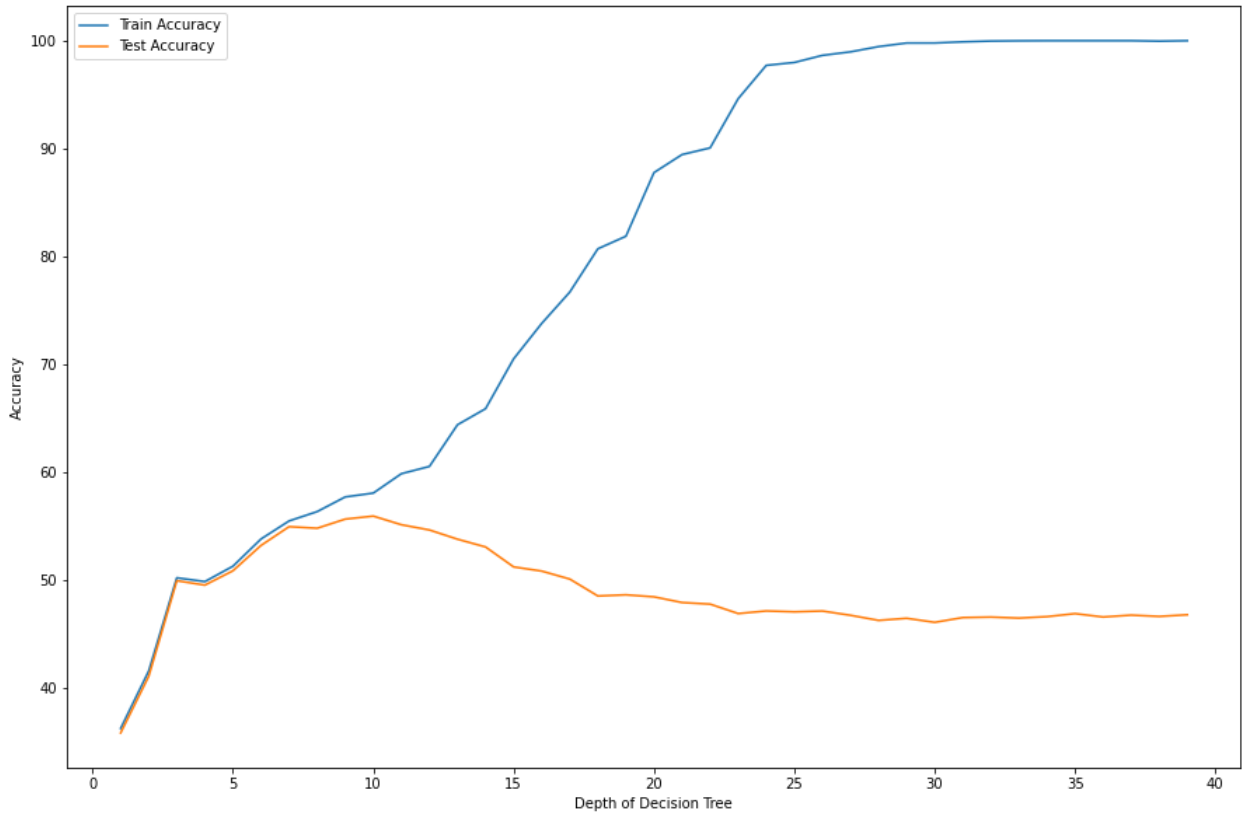


We can see that the accuracy increases with the maximum depth, up to a certain point, and then starts to decrease, indicating overfitting. ***The optimal depth for this dataset is around 7, where we achieve an accuracy of around 55.37%.***

For p3 dataset in A1:(Entropy Impurity)

Region	Depth	Train Accuracy	Test Accuracy
Underfitting	1	36.29	35.88
Underfitting	2	41.62	41.12
	3	50.25	49.98
	4	49.90	49.57
	10	58.09	55.96
	11	59.89	55.16
	12	60.56	54.68
	13	64.42	53.82
	18	80.72	48.56
	22	90.07	47.81
	25	98.64	47.17
Overfitting	27	98.96	46.78
Overfitting	28	99.45	46.30
Overfitting	39	100.0	46.82

Bias-Variance for entropy Impurity, Accuracy vs Depth for Entropy Impurity(For p3 dataset in A1)



For entropy, the sweet spot is around depth 10, after depth 10 the algorithm started overfitting. Best test accuracy was around 55% at depth 10.

***Inference** : For depth around 11, we are already overfitting, Hence it is very easy to over fit in decision trees.*

For PCA_MNIST dataset in A1:(Gini Impurity)

For depth 5 and minimum samples per split allowed at each node is 3.(This can also act as regularizer).

Classification Accuracy on Test Data is: 68.43

Confusion Matrix is:

[[1299.0, 0.0, 6.0, 5.0, 31.0, 0.0, 1.0, 74.0, 304.0, 80.0],

[22.0, 1102.0, 3.0, 8.0, 402.0, 3.0, 44.0, 88.0, 120.0, 8.0],

[95.0, 5.0, 1212.0, 249.0, 25.0, 76.0, 23.0, 79.0, 27.0, 9.0],

[23.0, 0.0, 35.0, 1475.0, 104.0, 4.0, 0.0, 131.0, 28.0, 0.0],

[0.0, 1.0, 46.0, 64.0, 1595.0, 15.0, 4.0, 43.0, 32.0, 0.0],

[5.0, 4.0, 13.0, 98.0, 24.0, 1495.0, 134.0, 2.0, 20.0, 5.0],

[47.0, 8.0, 30.0, 293.0, 30.0, 417.0, 924.0, 25.0, 19.0, 7.0],

[78.0, 1.0, 0.0, 11.0, 315.0, 1.0, 0.0, 1376.0, 12.0, 6.0],

[461.0, 2.0, 11.0, 63.0, 147.0, 20.0, 7.0, 50.0, 1019.0, 20.0],

[421.0, 1.0, 16.0, 21.0, 10.0, 6.0, 1.0, 473.0, 30.0, 821.0]]

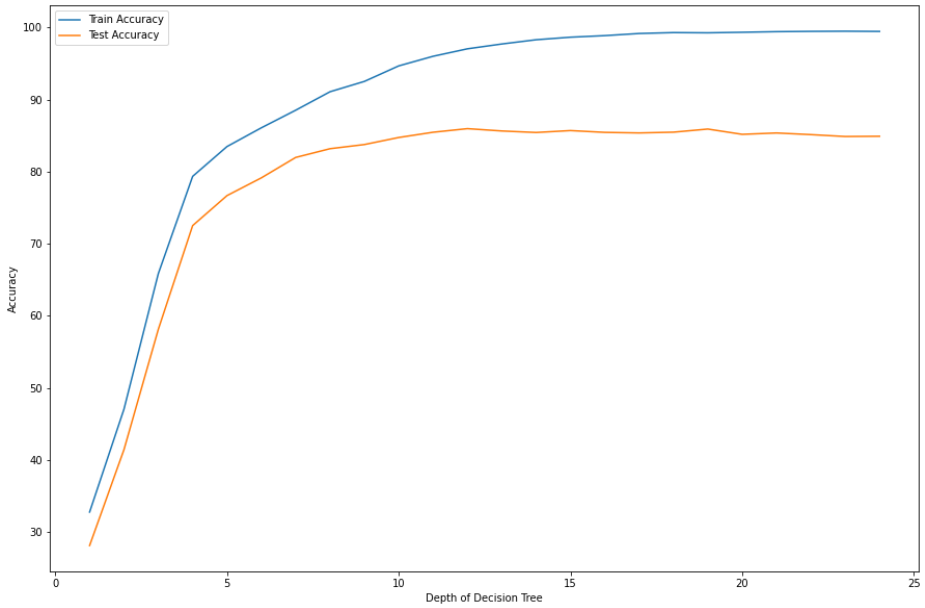
f1 Score of Class 1 is: 0.611150317572336
f1 Score of Class 2 is: 0.7537619699042409
f1 Score of Class 3 is: 0.764186633039092
f1 Score of Class 4 is: 0.7218008319060435
f1 Score of Class 5 is: 0.7115770689270577
f1 Score of Class 6 is: 0.7792546260099036
f1 Score of Class 7 is: 0.628999319264806
f1 Score of Class 8 is: 0.6645737744506158
f1 Score of Class 9 is: 0.5974787452360012
f1 Score of Class 10 is: 0.5957910014513789

For different depths:(Bias-Variance)

Region	Depth	Train Accuracy	Test Accuracy
Underfitting	1	32.77	28.13
Underfitting	2	47.04	41.42
	3	65.84	58.12
	4	79.34	72.51
	5	83.47	76.67
	6	86.07	79.13
	7	88.51	81.96
	8	91.07	83.17
	9	92.51	83.75
	10	94.66	84.73
	17	99.15	85.36
Overfitting	24	99.44	84.90

For PCA MNIST, It took a while to overfit as we have 60k data points compared to 10k in p3. Hence after depth around 20 it started overfitting. The accuracy was good compared to p3. This can be due to p3 having some complex and non-linear relationships that are difficult for the decision tree algorithm to capture.

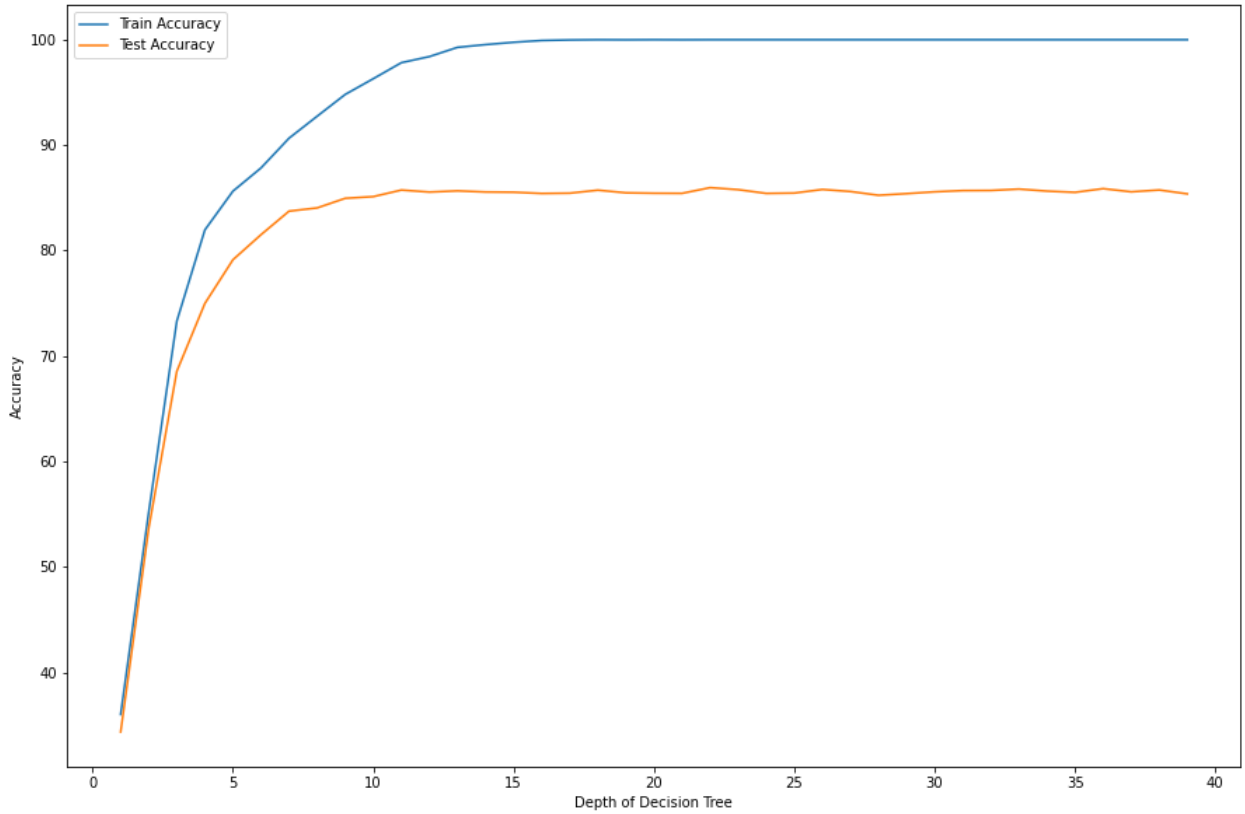
Bias-Variance for PCA_MNIST(Gini Impurity), Accuracy vs Depth for Gini Impurity(For PCA-MNIST dataset in A1)



For PCA_MNIST dataset in A1:(Entropy Impurity)

Region	Depth	Train Accuracy	Test Accuracy
Underfitting	1	36.0119	34.35
Underfitting	2	55.1976	53.6667
Underfitting	3	73.3048	68.55
	4	81.9571	74.9833
	5	85.6333	79.1278
	6	87.8357	81.5056
	7	90.6595	83.7389
	8	92.7405	84.0444
	9	94.8071	84.9556
	13	99.2738	85.6611
	14	99.5333	85.5556
	22	100.0	85.9667
	23	100.0	85.7778
	24	100.0	85.4222
	25	100.0	85.4556
	26	100.0	85.7944
	27	100.0	85.6
	36	100.0	85.8722
	37	100.0	85.5722
	38	100.0	85.7444
	39	100.0	85.3778

Bias-Variance for PCA_MNIST(Entropy Impurity), Accuracy vs Depth for Entropy Impurity(For PCA-MNIST dataset in A1)



Observations

- As the depth of the tree increases, both the train and test accuracies improve, indicating that the model is learning more complex patterns in the data. However, the test accuracy reaches a peak at a depth of 9 and then starts to decrease slightly. This suggests that the model is starting to overfit the training data and is not able to generalize well to new data.

- Increasing the depth of the tree further does not result in significant improvements in test accuracy and may even lead to a slight decrease in accuracy. This is likely due to the model becoming too complex and overfitting the training data even more.

- Overall, it seems that a decision tree model with a depth of 11 provides the best trade-off between complexity and accuracy for the PCA_MNIST dataset with entropy impurity.

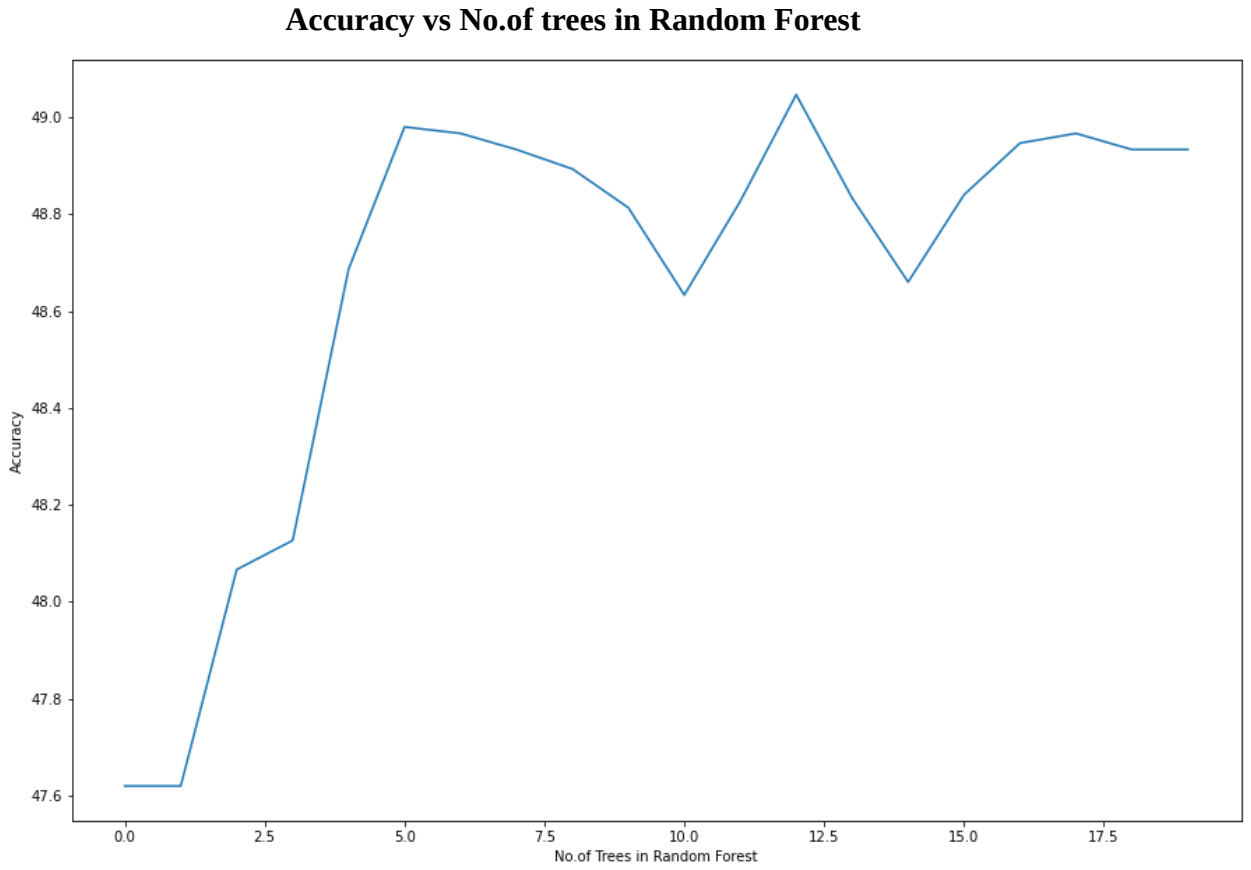
p2: Random Forests

Random Forests on p3 data:

Implemented random forests with varying number of trees from 1 to 20.

Accuracy scores for increasing trees from 1 to 20 in random forests are:

[47.62, 47.62, 48.07, 48.13, 48.69, 48.98, 48.97, 48.93, 48.89, 48.81, 48.63, 48.83, 49.05, 48.83, 48.66, 48.84, 48.95, 48.97, 48.93, 48.93]



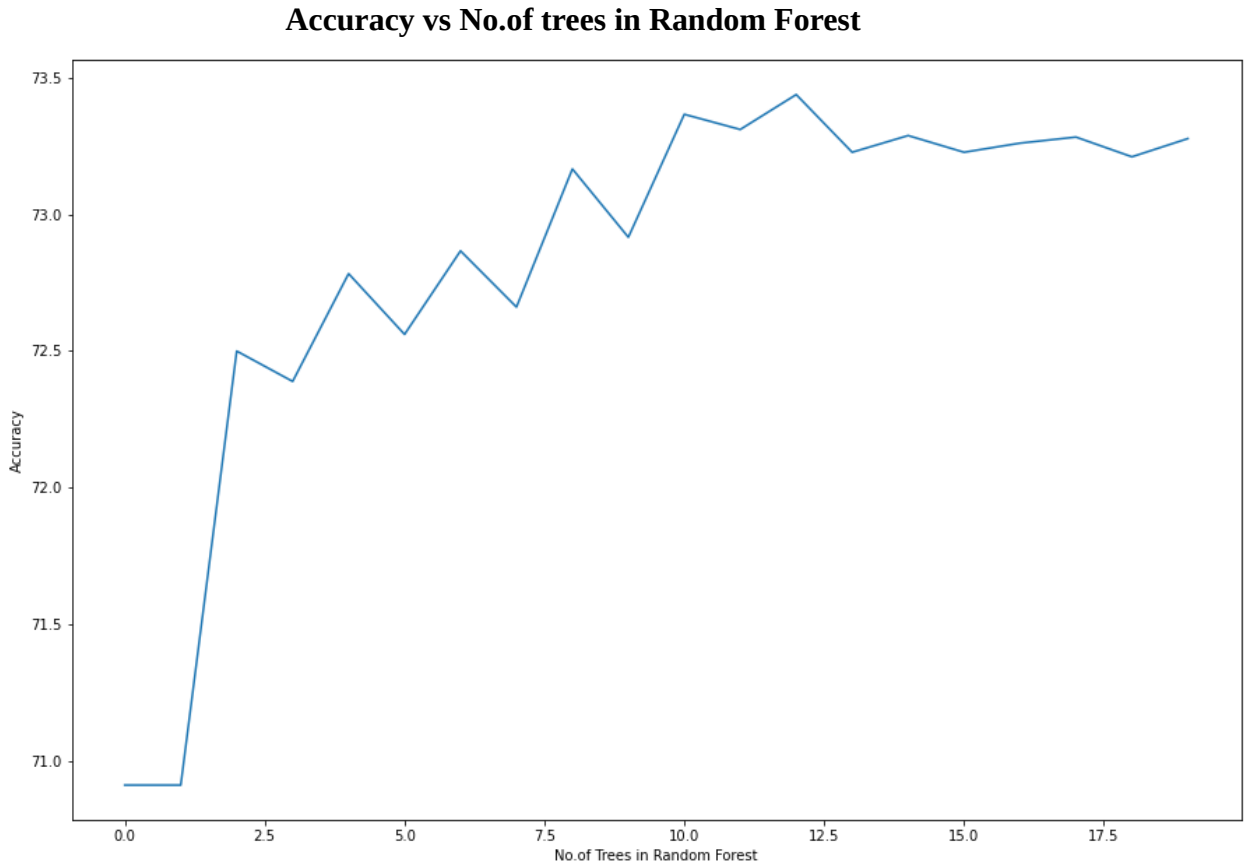
Observations

- These accuracy scores represent the performance of a random forest algorithm on a certain dataset with increasing numbers of trees in the forest.
- As the number of trees increases, the accuracy of the model generally improves, as the model becomes more complex and is able to capture more complex relationships in the data.
- However, in this specific case, the improvement in accuracy is not very significant as the number of trees increases, with the accuracy scores hovering around the range of 48-49%. This suggests that the model may be facing some limitations in capturing the underlying

patterns in the data, and increasing the number of trees beyond a certain point may not be very helpful.

Random Forests on PCA_MNIST Data:

Implemented random forests with varying number of trees from 1 to 20.



Observations

- The reported accuracy scores suggest that the random forest model is gradually improving in accuracy as the number of trees in the ensemble is increased. The initial accuracy score of 70.91% is obtained with just one tree in the forest, and this remains the same for the next iteration as well.
- However, as more trees are added to the ensemble, the accuracy score gradually improves. A noticeable increase in accuracy is observed when the number of trees is increased from 1 to 3, where the accuracy score jumps to 72.5%. Further increases in the number of trees lead to smaller improvements in accuracy, but the trend is generally upward.
- We can see that the accuracy increases with the number of trees, up to a certain point, and then levels off. This indicates that adding more trees beyond a certain point does not improve the accuracy significantly, but it increases the computation time and memory usage.

p3: Adaboost

Objective:

The objective of this problem is to implement Adaboost algorithm on various datasets using various types of classifiers as weak learners and observe its performance as the no. of weak learners increase.

Data:

PCA-MNIST data – 10 dim

Kannada-MNIST data – 784 dim

Assignment 1 p3 data – 10 dim

The given data is split into train and test data in 80:20 ratio (if test data is not explicitly given)

Architectures:

Ensemble – 1: Decision Trees of max depth 4. *A total of 10 weak learners*

Ensemble – 2: Alternate MLPs. 1 MLP with 16 neurons in 1 hidden layer. 1 MLP with 20 neurons in 1 hidden layer. 1 MLP with 30 neurons in 1 hidden layer. *A total of 30 weak learners*

Ensemble – 3: Alternate MLPs and Decision Trees. MLP with 1 hidden layer of 16 neurons. Decision Trees with max depth 4. *A total of 20 weak learners*

Algorithm 2 *SAMME*

1. *Initialize the observation weights* $w_i = 1/n,\ i = 1, 2, \ldots, n.$
2. *For* $m = 1$ *to* M*:*
   (a) *Fit a classifier* $T^{(m)}(\boldsymbol{x})$ *to the training data using weights* w_i.
   (b) *Compute*
   $$err^{(m)} = \sum_{i=1}^{n} w_i \mathbb{I}\left(c_i \neq T^{(m)}(\boldsymbol{x}_i)\right) / \sum_{i=1}^{n} w_i.$$
   (c) *Compute*
   $$\alpha^{(m)} = \log \frac{1 - err^{(m)}}{err^{(m)}} + \log(K - 1).$$
   (d) *Set*
   $$w_i \leftarrow w_i \cdot \exp\left(\alpha^{(m)} \cdot \mathbb{I}\left(c_i \neq T^{(m)}(\boldsymbol{x}_i)\right)\right),\ i = 1, \ldots, n.$$
   (e) *Re-normalize* w_i.
3. *Output*
   $$C(\boldsymbol{x}) = \arg\max_k \sum_{m=1}^{M} \alpha^{(m)} \cdot \mathbb{I}(T^{(m)}(\boldsymbol{x}) = k).$$

Reference: https://hastie.su.domains/Papers/samme.pdf

Implementation Details:

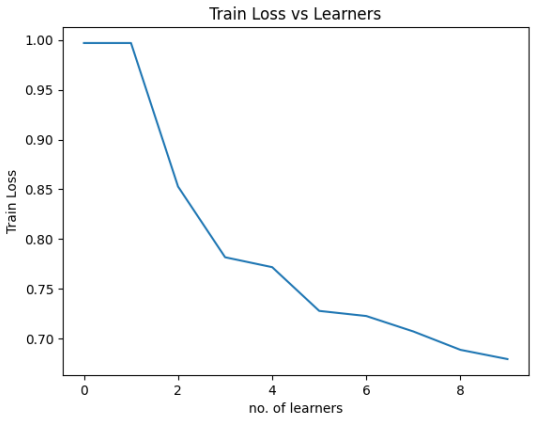
- Since MLPs are very powerful and can easily overfit the given data, In order to make them weak learners, we have only trained them for 2 epochs and also used sigmoid function instead of ReLU
- Inorder to avoid -ve step sizes, A variation of Adaboost algorithm (SAMME) is implemented

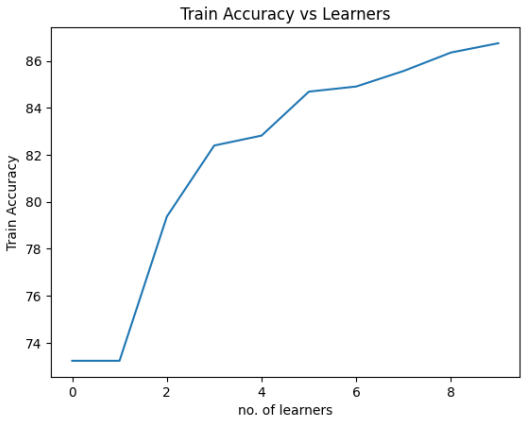
Results:

	PCA MNIST			Kannada MNIST			Assignment – 1 p3 Data		
Ensemble no.	Train Accuracy	Test Accuracy	Max.Weak Learner Accuracy	Train Accuracy	Test Accuracy	Max. Weak Learner Accuracy	Train Accuracy	Test Accuracy	Weak Learner Accuracy
1	86.75	89.16	82.68	54.67	54.5	49.59	-	-	-
2	85.24	88.25	56.26	-	-	-	85.55	88.34	56.04
3	88.05	90.50	70.36	-	-	-	54.95	55.22	49.9

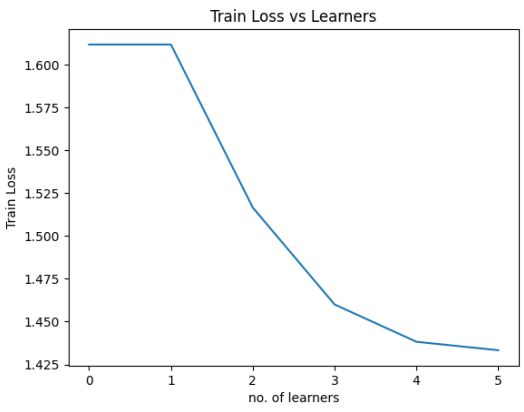
Kannada MNIST Data is not used for Ensemble models with Decesion trees as weak learners as they are computationally heavy

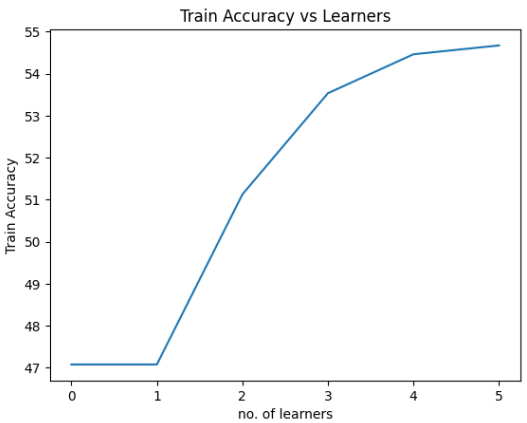
Ensemble – 1 (PCA- MNIST Data) Loss and Accuracy:



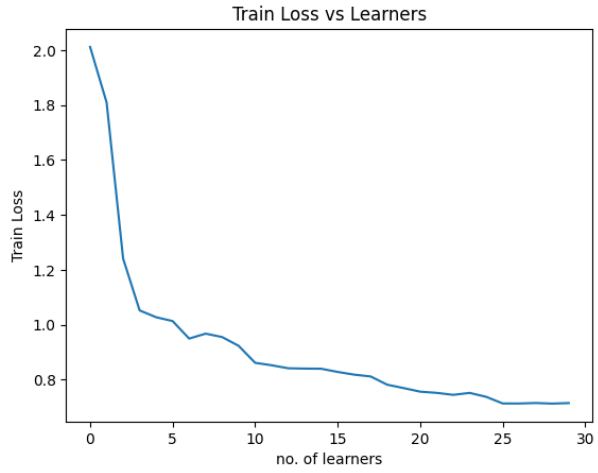


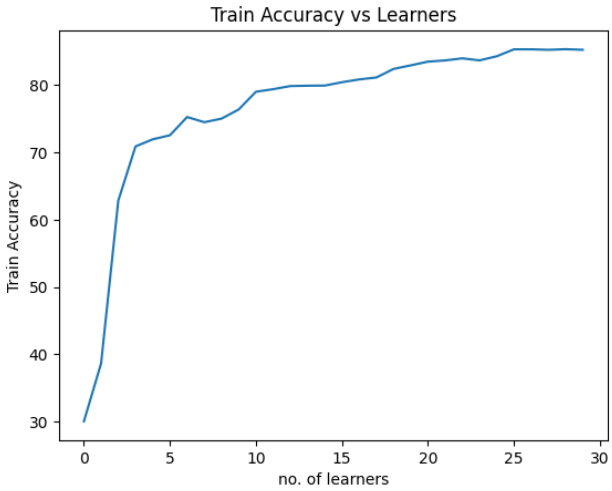
Ensemble – 1 (Assignment – 1 p3 Data) Loss and Accuracy:



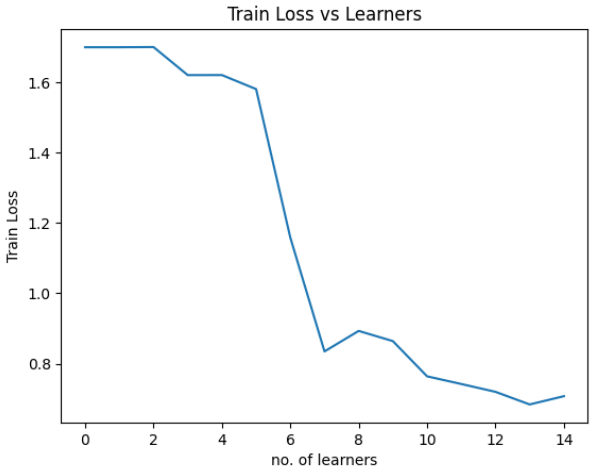


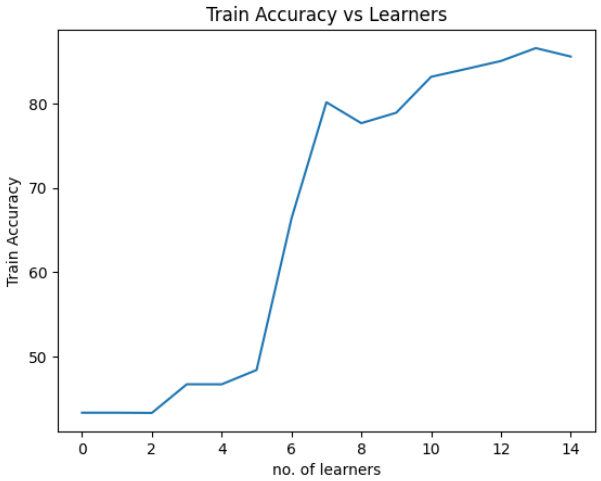
Ensemble – 2 (PCA - MNIST Data) Loss and Accuracy:



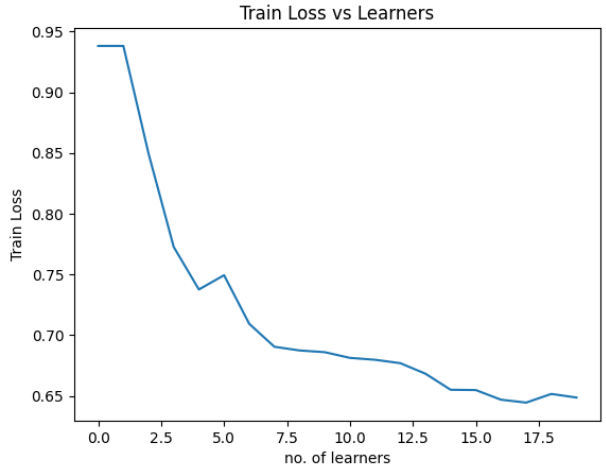


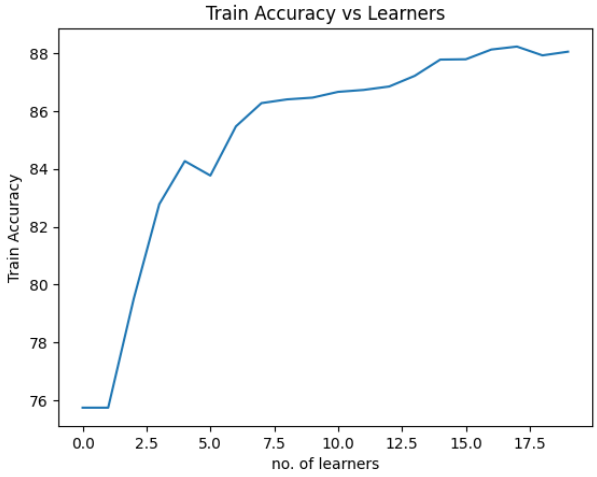
Ensemble – 2 (Assignment – 1 p3 Data) Loss and Accuracy:



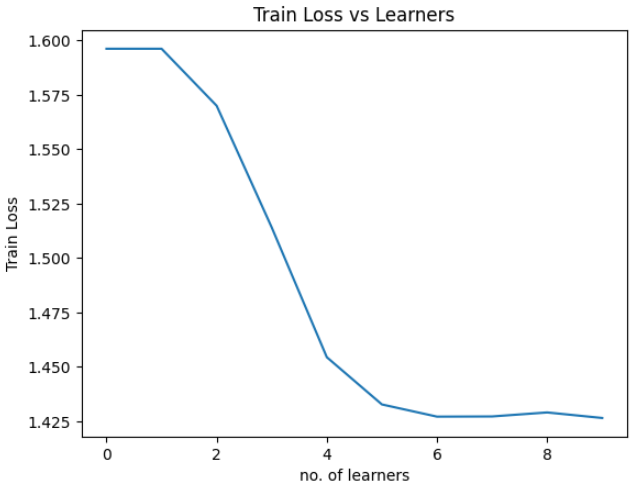


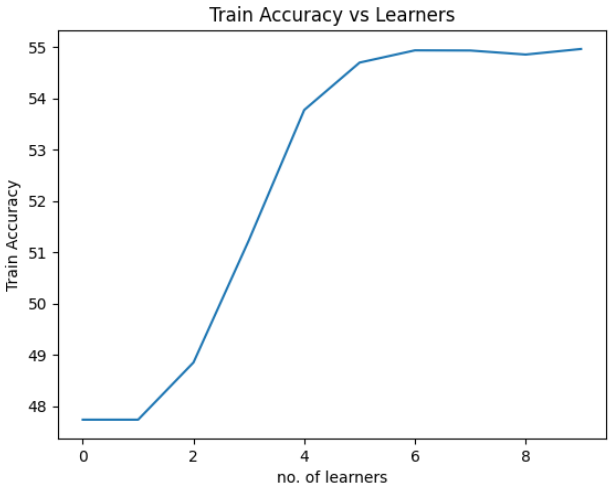
Ensemble – 3 (PCA – MNIST DATA) Loss and Accuracy:





Ensemble – 3 (Assignment – 1 p3 Data):





Observations:

- We can see that Maximum accuracy by any weak classifier is very less when compared to the accuracy of Ensemble model comprising those weak classifier
- The accuracy of MLPs trained for 2 epochs is 30% but when 30 of them are used in an Ensemble, it raised to 90% which shows how powerful Ensemble models are.

p4: KMeans and GMM Clustering

Objective:

The objective of this problem implement Kmeans and GMM Clustering algorithms on the given data and analyse the Normalized Mutual Information and t-SNE plots for various both algorithms with various no. of clusturs

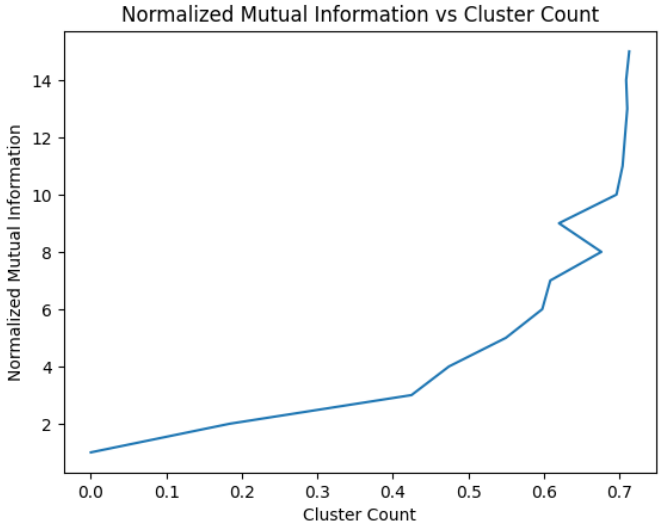
Data:

Kannada MNIST data is used for this problem

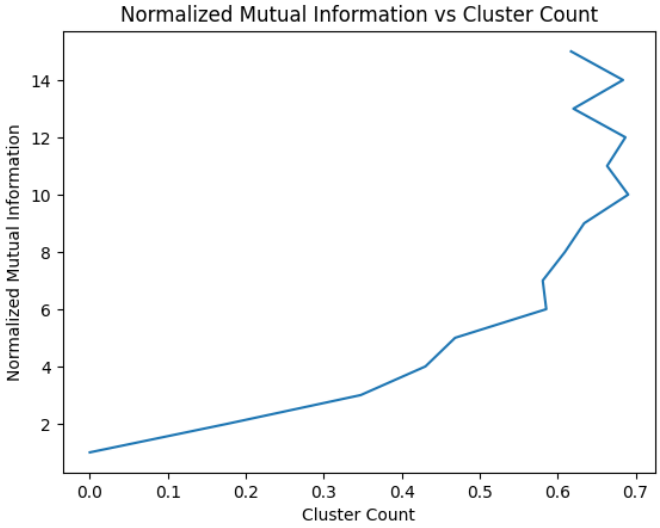
Results:

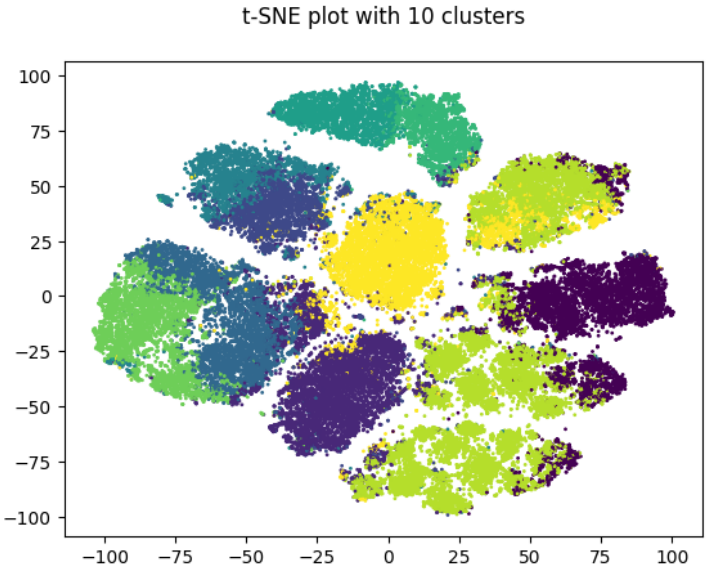
- We can see from the following plots that the Normalized Mutual Information increases with the increase in no. of clusters
- Both KMeans and GMM clustering are computationally very heavy

K Means Clustering

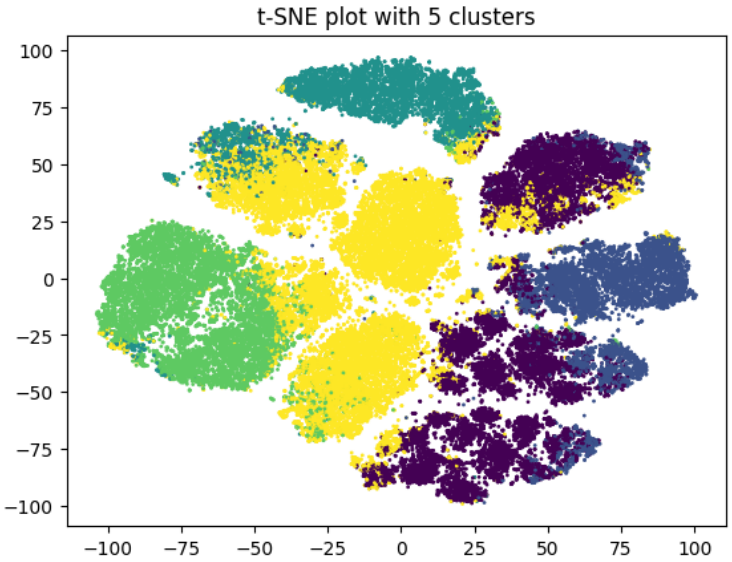


GMM based Clustering

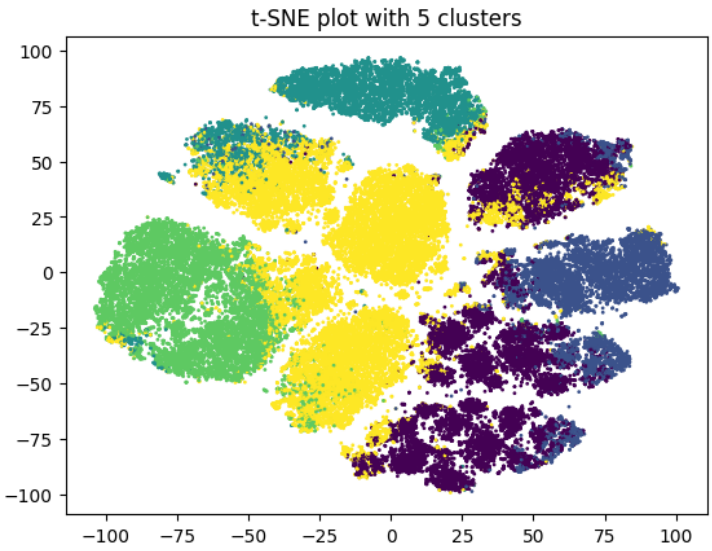




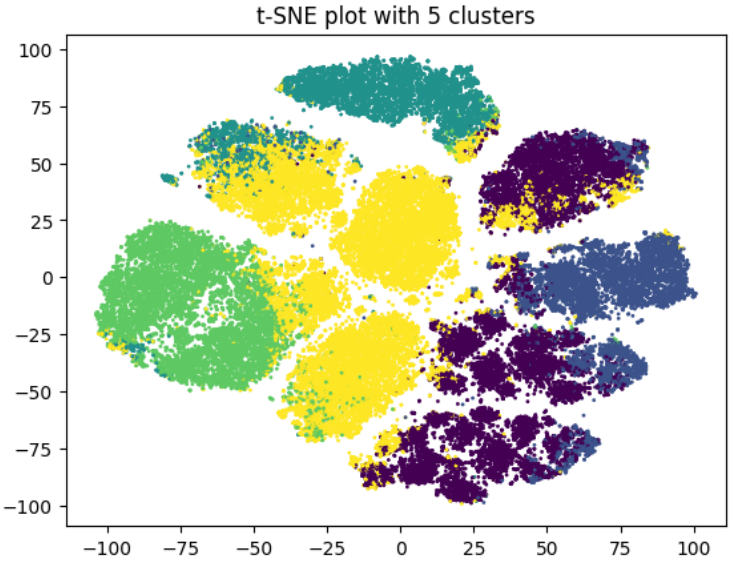
K Means with 10 clusters



K Means with 5 clusters



GMM clustering with 10 clusters

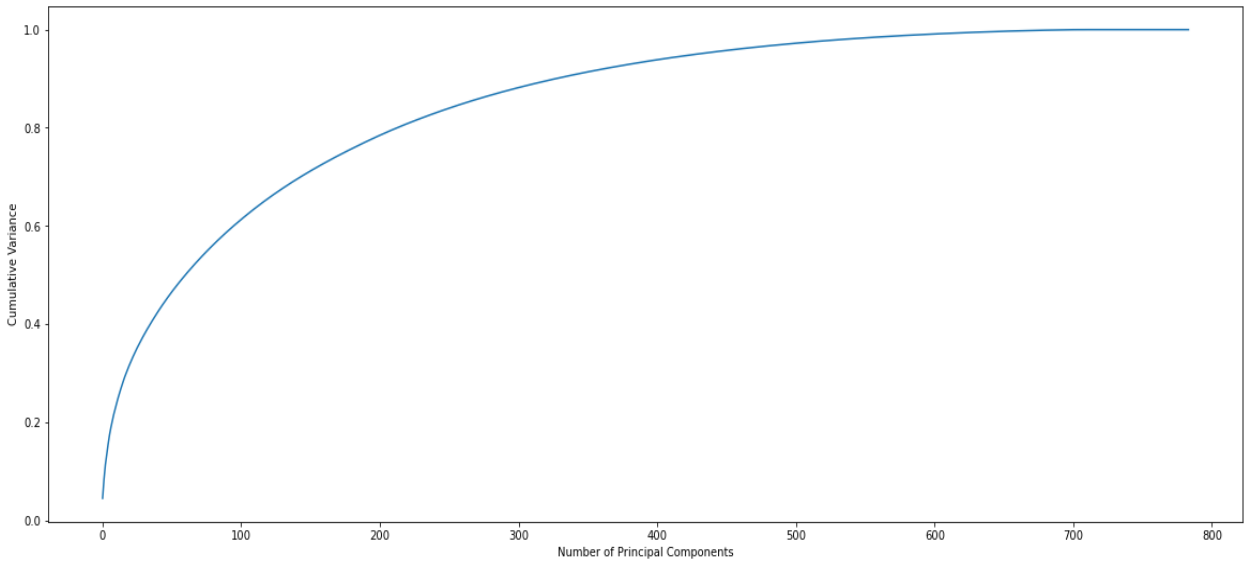


GMM clustering with 5 clusters

p5: PCA

Principal Component Analysis (PCA) is a popular technique for reducing the dimensionality of high-dimensional data while retaining most of the variation in the data. In this report, we implemented PCA on the KMNIST dataset and plotted the data variance as a function of the number of principal components.Note that KMNIST data points are in 784 dimensions.

Number of Principal components vs Cumulative variance



Observations

- We can see that the first few principal components explain a large amount of the variation in the data, with the first principal component explaining around **4.54%** of the variance, and the first 100 principal components explaining around **61.34%** of the variance.
- **For 95% variance, we need 429 components and for 90% we need 327 components.**
- **For 99% variance, we need 592 components**, Hence we can significantly reduce the dimensionality of the data by projecting it onto a smaller subspace spanned by the first few principal components.
- This indicates that we can significantly reduce the dimensionality of the data by projecting it onto a smaller subspace spanned by the first few principal components, without losing much information. **Even if we want to preserve 99% variance, we reduce the dimensions by 192, which is a significant reduction.**