Machine Learning Assignment 1

Reshan Faraz
PhD19006

Following are the Libraries used : The Library used as per the demand of the question if in question it is mentioned to not used library then i didn't use it.

1- pandas
2- numpy
3- sklearn
4 - tabulate
5 - plotly
6 - seaborn
7 - matplotlib
8 - scipy
9 - os

Preprocessing of data : For every question I explained the preprocessing of data in the respective answer part.

Answer 1

Preprocessing of Data : I load the data using sio.loadmat, the file is dictionary containing the `dict_keys(['__header__', '__version__', '__globals__', 'samples', 'labels']) (for both dataset_2.mat and dataset_1.mat).`

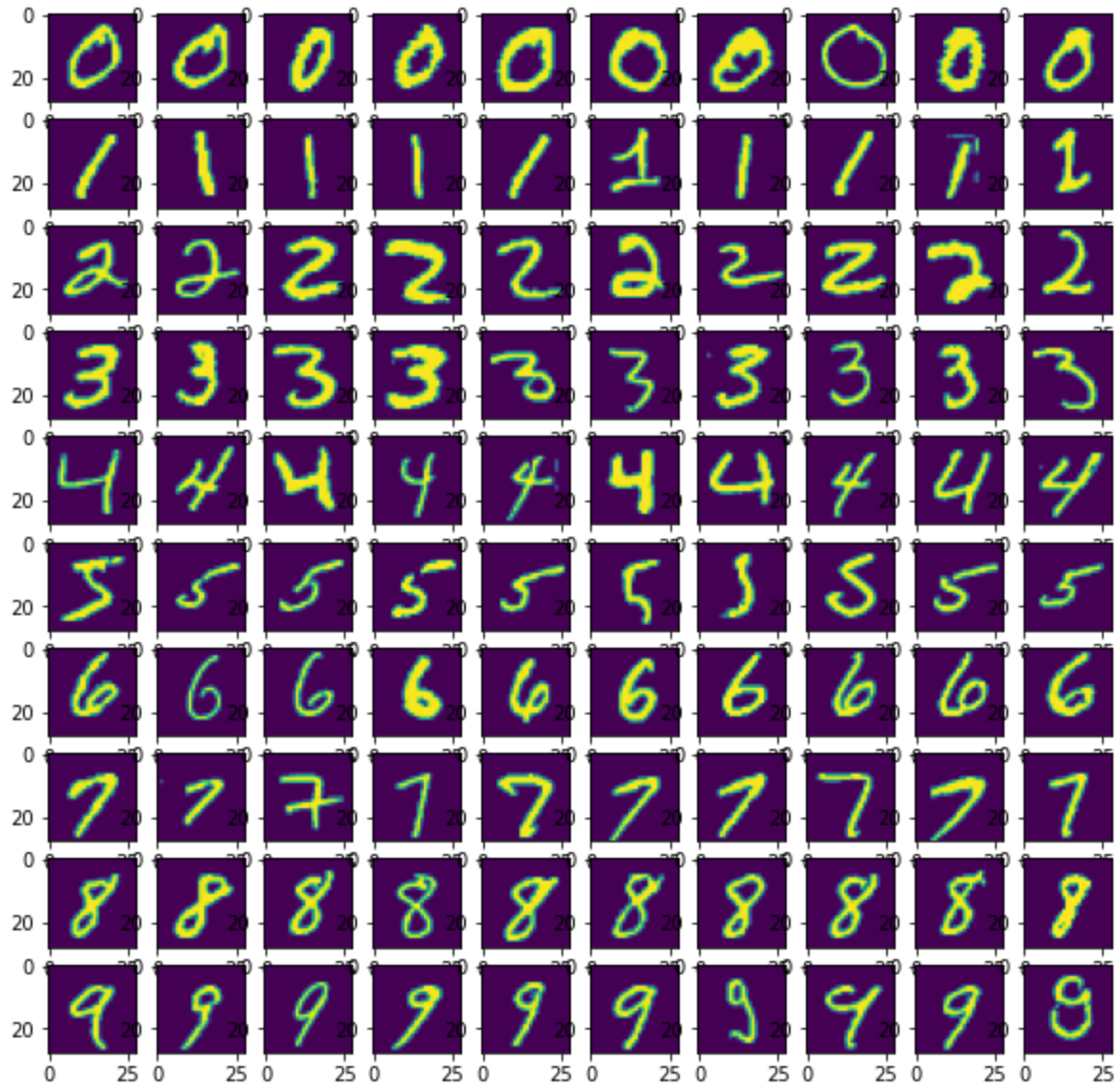
After that I create two numpy arrays one to contain the features and another to contain the labels and I select 10 samples from each label by using simple for loop for the visualization part.

dataset_1.mat contain samples of size =(50000,28,28) and labels of size =(1,50000) after converting to numpy array

dataset_2.mat contain samples of size =(20000,2) and labels of size =(1,20000) after converting to numpy array

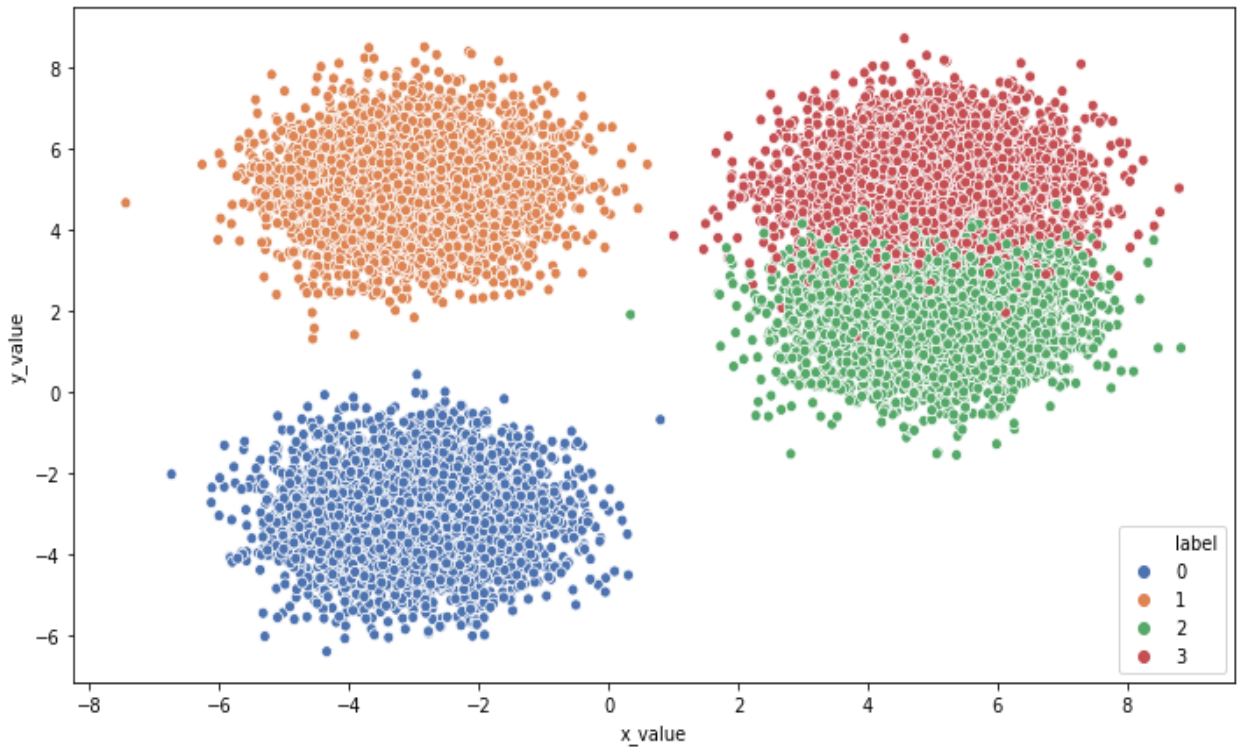
Part a :)

Following are my observations with each first 10 samples of each class .I plot the graph using matplotlib and observe that it is the mathematical integer value from 0-9 . The graph contains the 10 samples from each label.



Part b:

For the data visualization I used seaborn to plot the scatter plot ,following are the result:



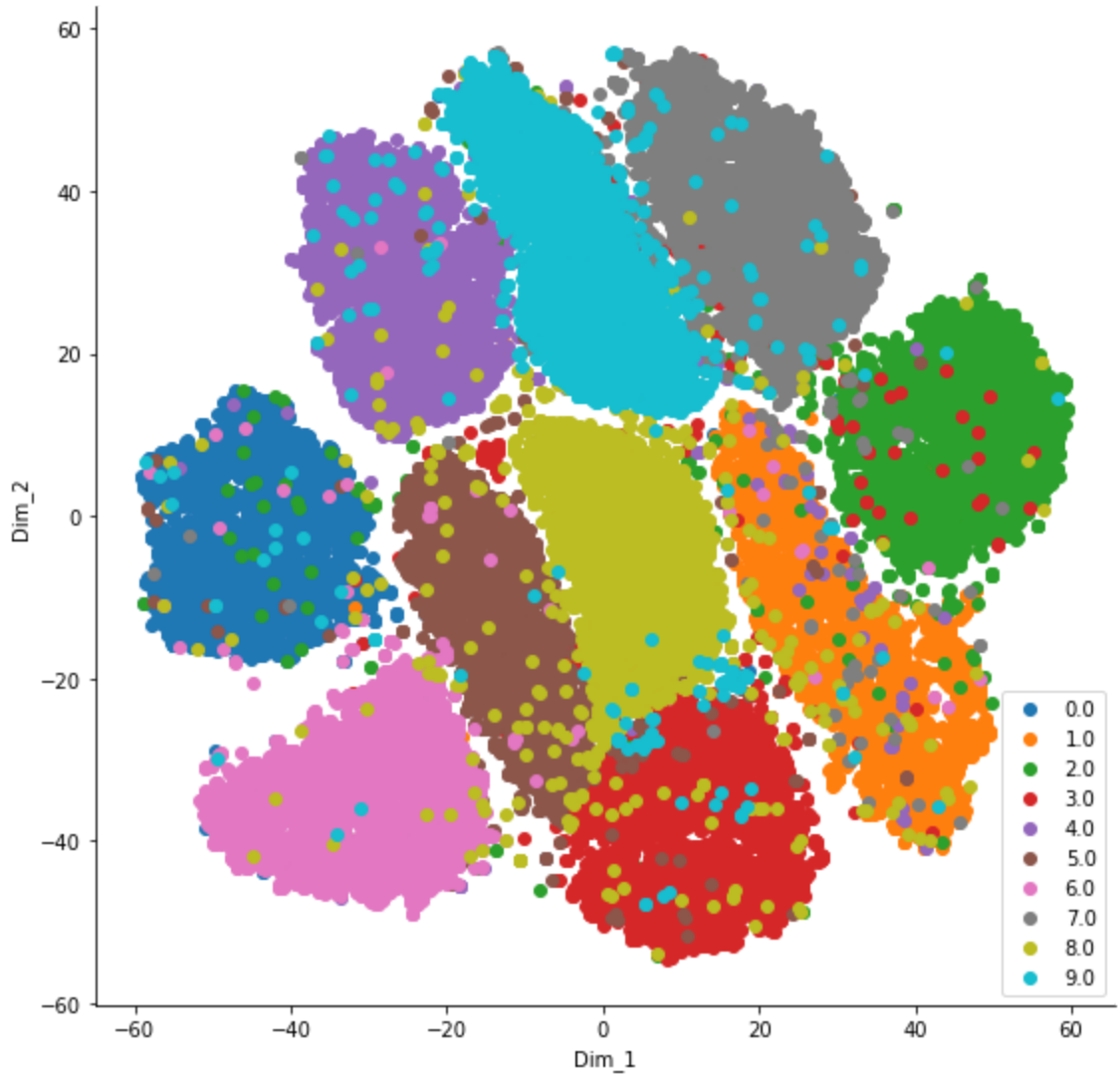
Inference about the data set : From the above graph it is clear that class of label 0 and 2 are easily separable while there is overlapping between label 2 and 3. Label 2 and 3 are very close to each other means they are having few features common.

Part c:

t-SNE is used to reduce the dimension for this part I reduce the dimension to 28*28 = (784) feature to 2 Following are parameter set while using TSNE `n_components = 2, random_state = 0,perplexity=30,n_iter=1000`

The perplexity and number of iterations plays an important parameter while reducing the dimensionality

The inference regarding the class separation from the graph is that classes are well separated even when we reduce the dimension of the feature these classes are the Integer from 0 -9 .



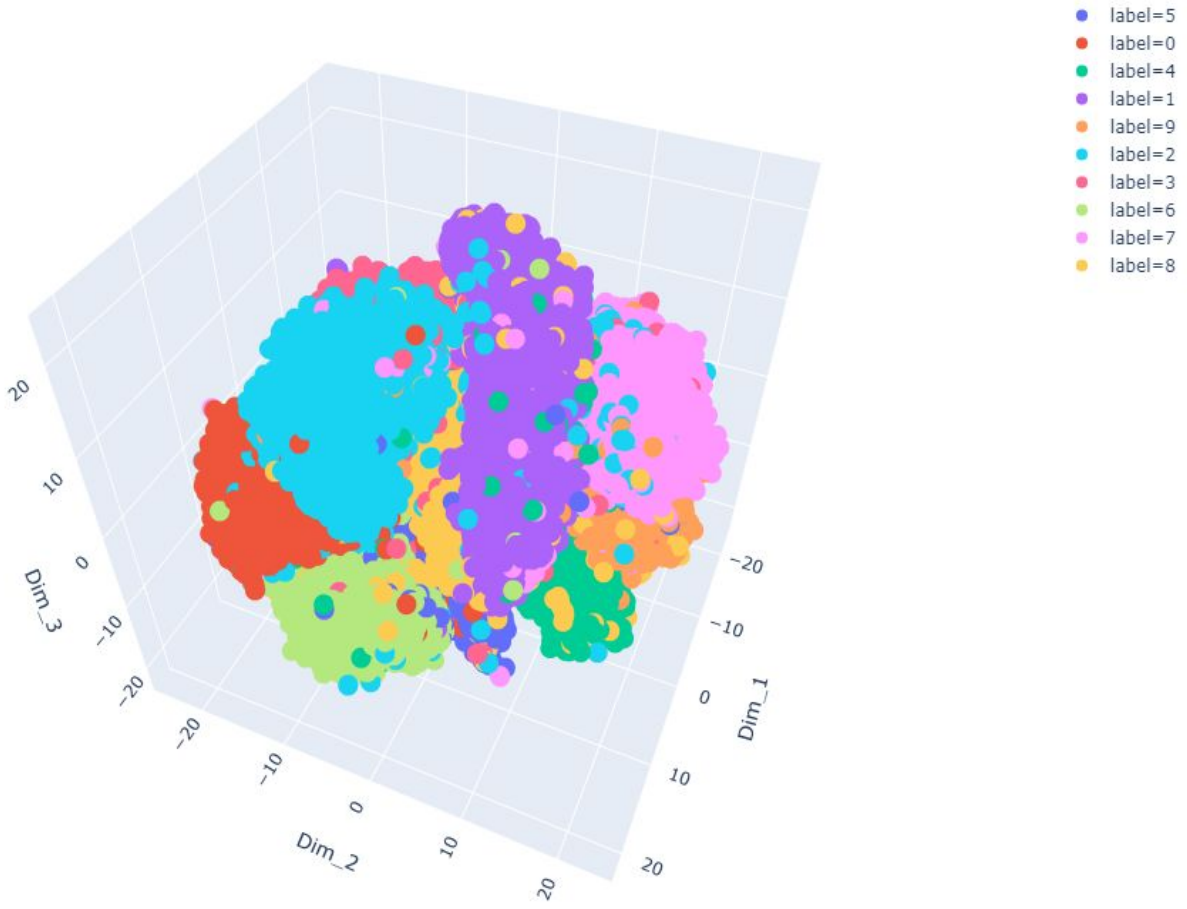
Part d)

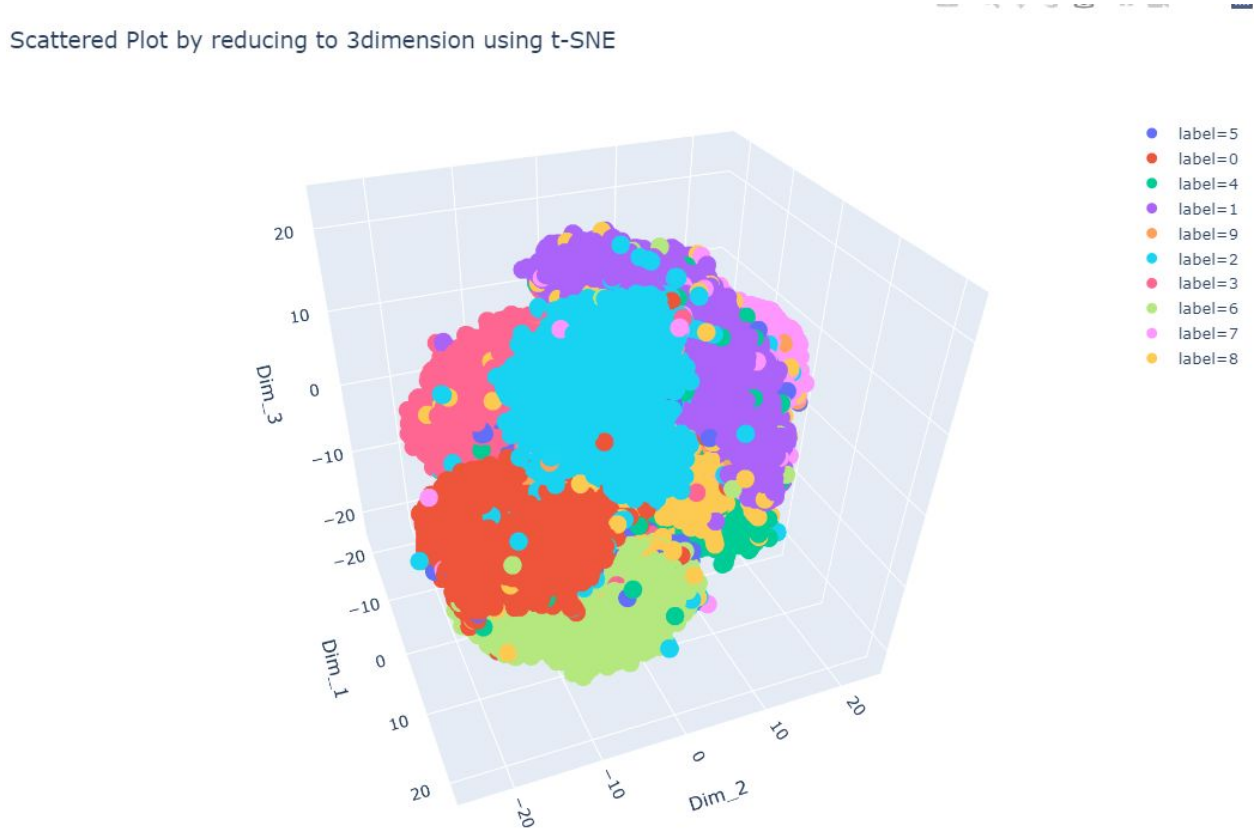
t-SNE is used to reduce the dimension for this part I reduce the dimension to 28*28 = (784) feature to 3 Following are parameter set while using TSNE `n_components = 2, random_state = 0,perplexity=30,n_iter=1000`

The perplexity and number of iterations plays an important parameter while reducing the dimensionality

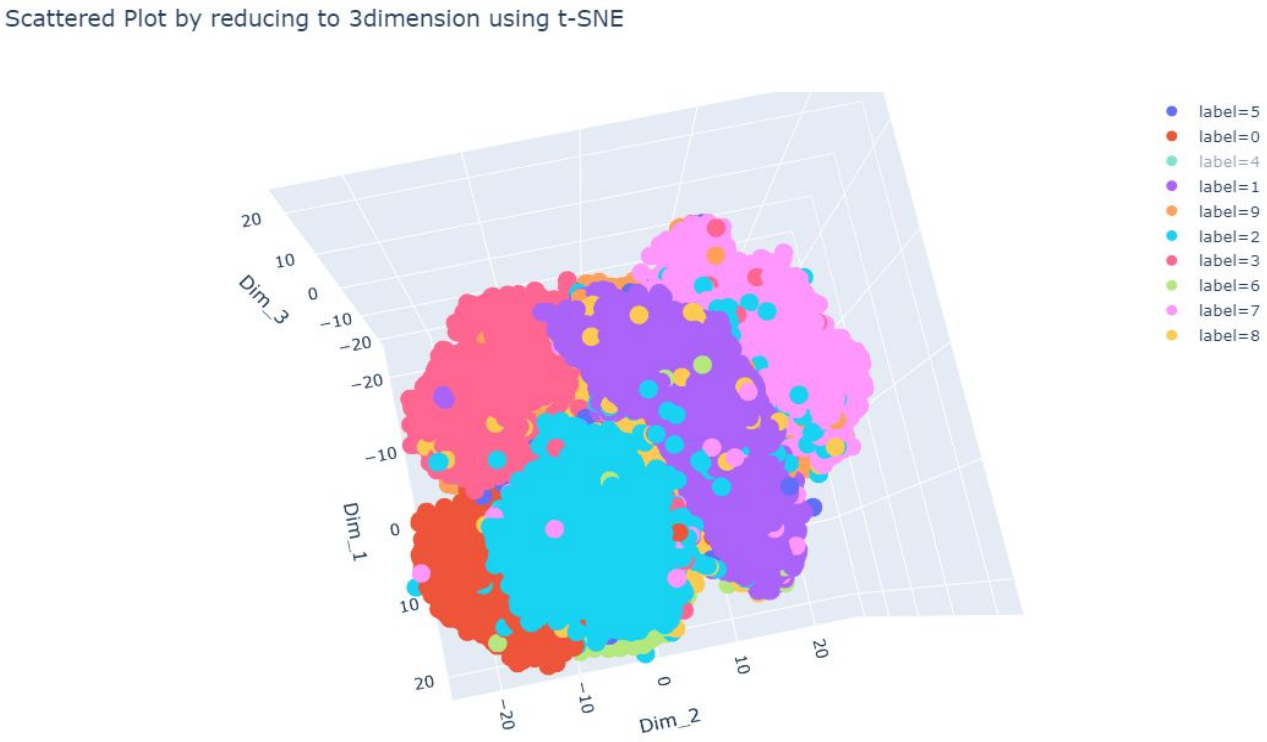
The inference regarding the class separation from the graph is that if we have a large dimension which can not be visualized easily we can reduce the dimension and then visualize it. For this part i attached two graph for better visualization.

Scattered Plot by reducing to 3dimension using t-SNE




Scattered Plot by reducing to 3dimension using t-SNE
label=5
label=0
label=4
label=1
label=9
label=2
label=3
label=6
label=7
label=8
Dim_3
Dim_1
Dim_2
20
10
0
−10
−20

Top view:



Answer 2

Preprocessing of Data:

I load the data using sio.loadmat, the file is dictionary containing the `dict_keys(['__header__', '__version__', '__globals__', 'samples', 'labels']) (for both dataset_2.mat).`

After that I create two numpy arrays one to contain the features and another to contain the labels and I select 10 samples from each label by using simple for loop for the visualization part.

dataset_2.mat contain samples of size =(20000,2) and labels of size =(1,20000) after converting to numpy array

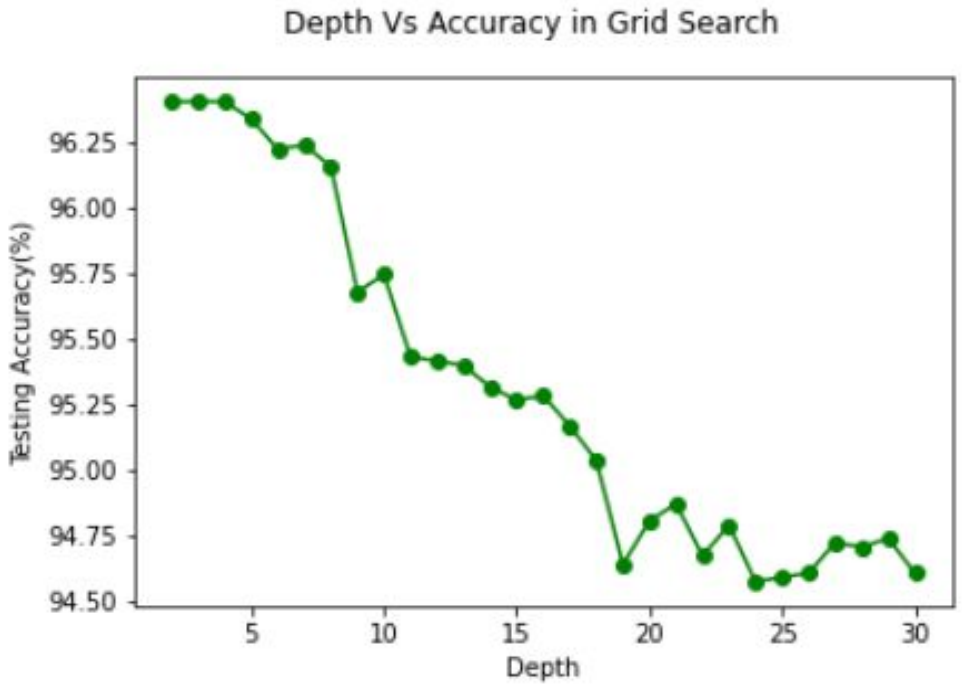
For splitting the data set into 70% of training set and 30% testing set I used **numpy randn** to generate uniformly distributed random number between [0,1] and then apply <0.7 comparison element wise to store the result into training set and rest in testing set.

Part a)

I performed **grid search** for values between 2 to 30 of depth here the depth is my hyperparameter .Observation is that with depth increases the Testing Accuracy will be slightly decreased ,this may be due to the overfitting of the model. I split the data into 70 and 30 ratio using uniform distribution.Yes the result is consistent with the part 1-b answer as there are only four class as Testing accuracy is max till depth 4. Following are graph attached which show the performance of grid search with increase in depth and also the output.

The optimal value is for depth 2,3,4 with Testing accuracy **96.40382711976245**

Depth 2 Testing Accuracy 96.40382711976245
Depth 3 Testing Accuracy 96.40382711976245
Depth 4 Testing Accuracy 96.40382711976245
Depth 5 Testing Accuracy 96.33784229627186
Depth 6 Testing Accuracy 96.22236885516331
Depth 7 Testing Accuracy 96.23886506103597
Depth 8 Testing Accuracy 96.15638403167272
Depth 9 Testing Accuracy 95.67799406136588
Depth 10 Testing Accuracy 95.74397888485649
Depth 11 Testing Accuracy 95.43055097327614
Depth 12 Testing Accuracy 95.4140547674035
Depth 13 Testing Accuracy 95.39755856153084
Depth 14 Testing Accuracy 95.3150775321676
Depth 15 Testing Accuracy 95.26558891454965
Depth 16 Testing Accuracy 95.28208512042231
Depth 17 Testing Accuracy 95.16661167931376
Depth 18 Testing Accuracy 95.03464203233257
Depth 19 Testing Accuracy 94.63873309138899
Depth 20 Testing Accuracy 94.80369515011547
Depth 21 Testing Accuracy 94.86967997360607
Depth 22 Testing Accuracy 94.67172550313428
Depth 23 Testing Accuracy 94.78719894424282
Depth 24 Testing Accuracy 94.57274826789839
Depth 25 Testing Accuracy 94.58924447377103
Depth 26 Testing Accuracy 94.60574067964369
Depth 27 Testing Accuracy 94.72121412075222
Depth 28 Testing Accuracy 94.70471791487958
Depth 29 Testing Accuracy 94.73771032662488
Depth 30 Testing Accuracy 94.60574067964369

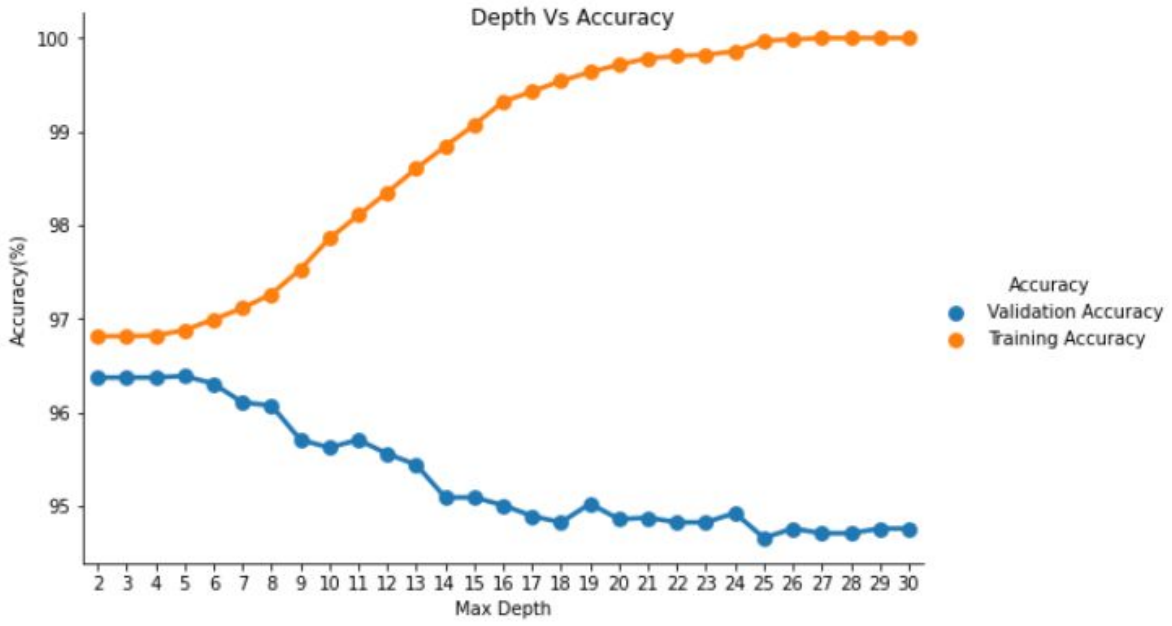


Part b):

I took the maximum depth value from 2 to 30 which produces the following result.
I used tabulate library to print the data int table format here is the snapshot
From the below table it is clear that for the max depth 2-4 our Validation Accuracy and Training Accuracy almost same and from max depth 5 to 25 training accuracy will increase and validation accuracy will decrease means that our model will try to **overfit the training data**. For the max depth 26-30 out training accuracy is 100% means our and validation accuracy is almost same, so here the model **overfit the training the data.**

Max Depth	Validation Accuracy	Training Accuracy
2	96.6968	96.6087
3	96.6968	96.6087
4	96.6968	96.6372
5	96.6298	96.7156
6	96.5627	96.8367
7	96.5459	97.0077
8	96.4118	97.1502
9	96.3112	97.371
10	96.3615	97.5777
11	96.1938	97.7558
12	96.0429	98.0764
13	95.9759	98.283
14	95.7579	98.5466
15	95.389	98.8102
16	95.3387	99.0809
17	95.3052	99.2662
18	95.171	99.4799
19	95.1207	99.601
20	95.0201	99.7435
21	95.0201	99.8076
22	95.0369	99.8575
23	95.0201	99.9288
24	94.8692	99.9644
25	94.9195	99.9929
26	94.9531	100
27	94.8524	100
28	94.8692	100
29	94.8357	100
30	94.9531	100

Here is the Graph Attached for better visualization.



Part c :

When I am trying to find the Accuracy using my function implementation and from the sklearn library both results are the same and it is true for both the Validation Accuracy and Training Accuracy . For accuracy I find the total number of matches from our model prediction and take the percentage. Here is the tabulated form of result both for Validation and Training Accuracy.

Max Depth	Validation Accuracy User Define	Valdation Accuracy sklearn
2	96.4268	96.4268
3	96.4268	96.4268
4	96.4268	96.4268
5	96.4103	96.4103
6	96.0463	96.0463
7	96.129	96.129
8	95.9801	95.9801
9	95.9801	95.9801
10	96.0629	96.0629
11	95.9305	95.9305
12	95.732	95.732
13	95.7155	95.7155
14	95.517	95.517
15	95.5335	95.5335
16	95.3515	95.3515
17	95.2026	95.2026
18	95.0703	95.0703
19	94.9545	94.9545
20	95.0372	95.0372
21	94.938	94.938
22	94.8387	94.8387
23	94.8718	94.8718
24	94.9711	94.9711
25	94.8056	94.8056
26	94.9049	94.9049
27	94.9876	94.9876
28	94.7891	94.7891
29	94.8883	94.8883
30	94.8883	94.8883

Max Depth	Training Accuracy User Defined	Training Accuracy sklearn
2	96.761	96.761
3	96.761	96.761
4	96.761	96.761
5	96.7968	96.7968
6	96.9545	96.9545
7	97.1265	97.1265
8	97.2841	97.2841
9	97.5134	97.5134
10	97.7786	97.7786
11	98.0294	98.0294
12	98.3303	98.3303
13	98.5453	98.5453
14	98.8105	98.8105
15	99.0111	99.0111
16	99.2261	99.2261
17	99.4052	99.4052
18	99.5629	99.5629
19	99.656	99.656
20	99.7635	99.7635
21	99.8638	99.8638
22	99.8997	99.8997
23	99.914	99.914
24	99.957	99.957
25	99.9642	99.9642
26	99.9785	99.9785
27	99.9785	99.9785
28	99.9857	99.9857
29	99.9928	99.9928
30	99.9928	99.9928

Answer 3

Preprocessing of data : I read the csv file using pandas library after that I remove the column which is not over interest or useless as the column ‘No’ is the just row index. After that there are many NAN values in the column I fill the NAN with the median of the column. The target variable is ‘Month’

I split the data into testing and training in the 80% and 20%. for slitting the data i used numpy randn to generate the uniformly distributed random number that split the data.

Part a):

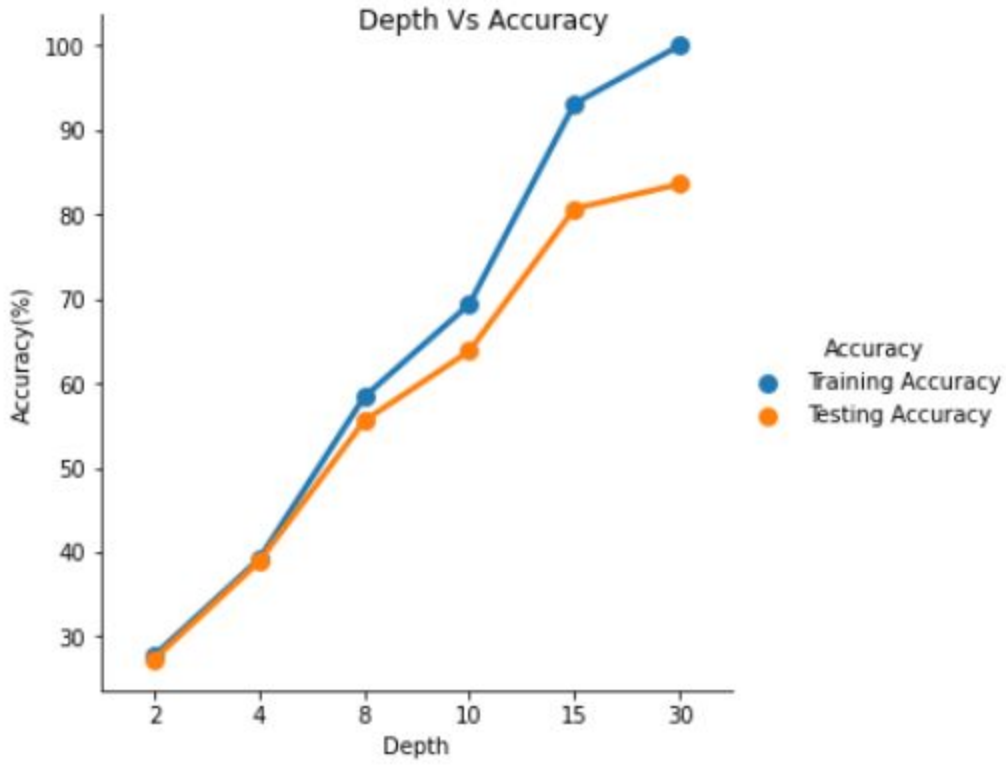
I used DecisionTreeClassifier from sklearn to train our model and then used an accuracy metric to calculate the score. Here is the result for the same.

Accuracy: using Gini Index **0.8373038274810884**
Accuracy: using Entropy **0.8414813142147454**
Best is Entropy

Part b): The best value of depth by using testing and training accuracy is easily obtained from the below table which is **30** having training accuracy 100 and Testing accuracy is **82.97608303249098**

Depth	Training accuracy	Testing Accuracy
2	27.414187643020593	28.068592057761734
4	39.78260869565217	40.03835740072203
8	59.31636155606407	57.682761732851986
10	70.51201372997711	66.14395306859205
15	93.94450800915331	80.49413357400722
30	100.0	82.97608303249098

Here is the graph attached for better visualization and analysis.



Part c):

For this part I divide my data in 80% and 20% as training and testing data and further divide the 80% of training data by taking 50% random data for each stumps.
I predicted the test sample label using majority vote ,for each tree stumps
The accuracy is 35.5201% which is very low as compared to the answer of part 'a' because maximum depth of the part 'a' is very large as compared to 3 in c. Also for part b the accuracy is low the reason might be the low training data and only taking the majority vote as an accuracy measure .

Part d)

Max_depth =[4, 8, 10, 15, 20 ,30(best in part b)] and for number of trees I used trees=**[20,50,100,150,200,250,300,350]** .using majority voting for final prediction the result is following :I print the result using tabulate in the table format for analysis :We got the best Training Accuracy for depth **30** and number of tree =**350** having **99.9829** and Best Testing Accuracy for depth **20** and number of trees = **350** having **92.8887**, with increase in depth and number of trees the training accuracy is increasing which might lead to the overfitting of data while for testing accuracy it increase for depth 20 and trees 350 after that it decreases for depth 30 and increases with increase in number of trees.

Max Depth	No. of Trees	Training Accuracy	Testing Accuracy
4	20	40.8604	39.7868
4	50	40.8061	39.8435
4	100	40.3834	39.4805
4	150	40.7033	39.6734
4	200	40.5805	39.7528
4	250	40.5833	39.6393
4	300	40.6662	39.7414
4	350	40.589	39.662
8	20	64.7013	61.7444
8	50	65.6554	62.8785
8	100	66.0496	63.094
8	150	66.3924	63.4343
8	200	66.441	63.491
8	250	66.1696	63.094
8	300	66.4639	63.3662
8	350	66.6695	63.491
10	20	79.3898	74.6966
10	50	79.7412	74.9688
10	100	80.3382	75.638
10	150	80.9324	76.1597
10	200	80.9724	76.5113
10	250	81.1781	76.3865
10	300	81.0638	76.3752
10	350	80.861	76.103
15	20	97.9861	89.4976
15	50	98.6088	89.8718
15	100	98.626	90.2688
15	150	98.806	90.7338
15	200	98.8888	90.5183
15	250	98.8288	90.7111
15	300	98.9374	90.7678
15	350	98.9374	90.8586
20	20	99.6201	91.403
20	50	99.9	91.868
20	100	99.9457	92.4804
20	150	99.9743	92.4351
20	200	99.98	92.6619
20	250	99.9714	92.5485
20	300	99.9743	92.3897
20	350	99.9714	92.8887
30	20	99.6772	91.006
30	50	99.8886	91.8793
30	100	99.9486	92.299
30	150	99.9486	92.6846
30	200	99.9629	92.7753
30	250	99.9714	92.5598
30	300	99.98	92.8207
30	350	99.9829	92.5371

Analysis of Models on Testing data:
For all the above models I am getting the best model with depth 20 with decision tree stumps of number of trees =350. For this model i used Gini index as criterion the Accuracy is 92.887
For the others the rank are as follows
1-Gini Index with depth 20 having decision stumps (Ensembling)is **92.887**
2-Only using Entropy as Parameter **84.14813142147454**
3- Only Gini Index as parameter **83.73038274810884**
4- Using Gini Index with depth = 30 is **82.976**
5-Ensembling with number of trees =100 and depth = 3 is **35.5201**

As from the above different models accuracy it is clear that with increase in depth the accuracy increases. For the worst model there is very low accuracy as compared to others models this is because we are training the model with less amount of data and low number of depth. Apart from only increasing the depth we can also ensemble the decision stumps with different numbers of trees and achieve better accuracy as trying to ensemble with different depth and different numbers of trees ,I increase the accuracy around 8-10%.
