Details of Question 1

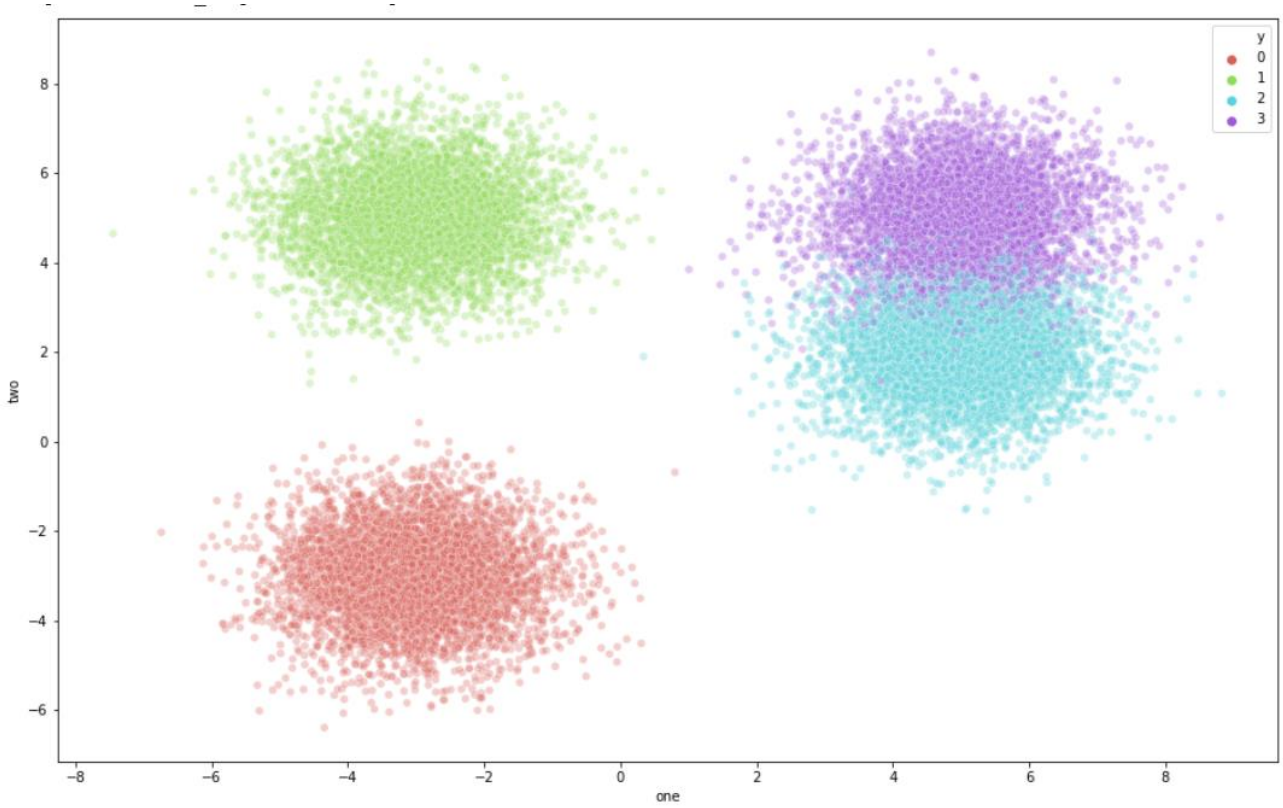
1(a) 10 samples of each class from 'dataset1' in the form of images

Observations: The dataset contains images of digits from 0-9. After loading it in a dictionary, it shows samples as features and labels as classes. There are 50,000 numbers of rows. Each row contains a 2D image (in the form of matrix) representing an image of a digit. Each corresponding column represents its class (i.e.; digit in integer value).

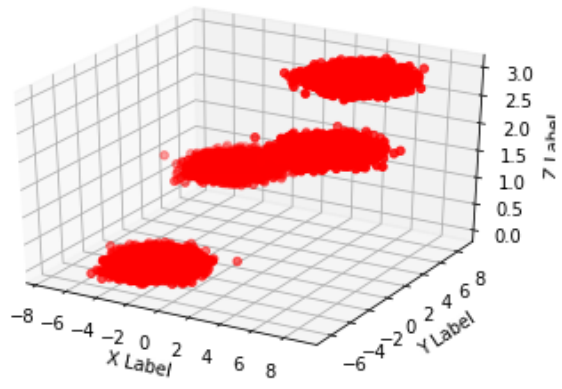
Images are in 1a_images.pdf file attached in this folder.

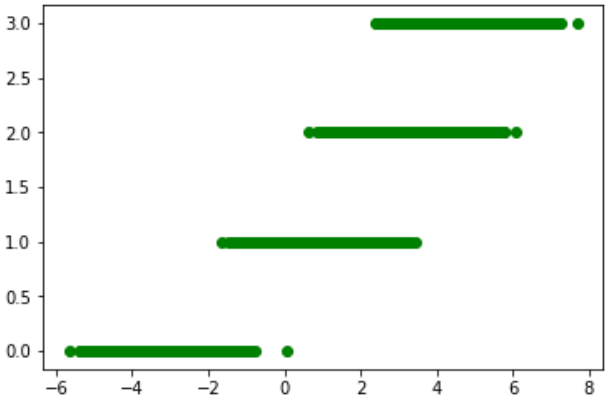
1(b) Visualization of the 'dataset2' in the form of scatter plot

Observation: After loading it in a dictionary, it shows samples as features and labels as classes. There are 20,000 numbers of rows. Each row contains 2 feature values. Each corresponding column represents its class (i.e.; one of (0,1,2,3)).
There seems to be four clusters having points in some ranges.



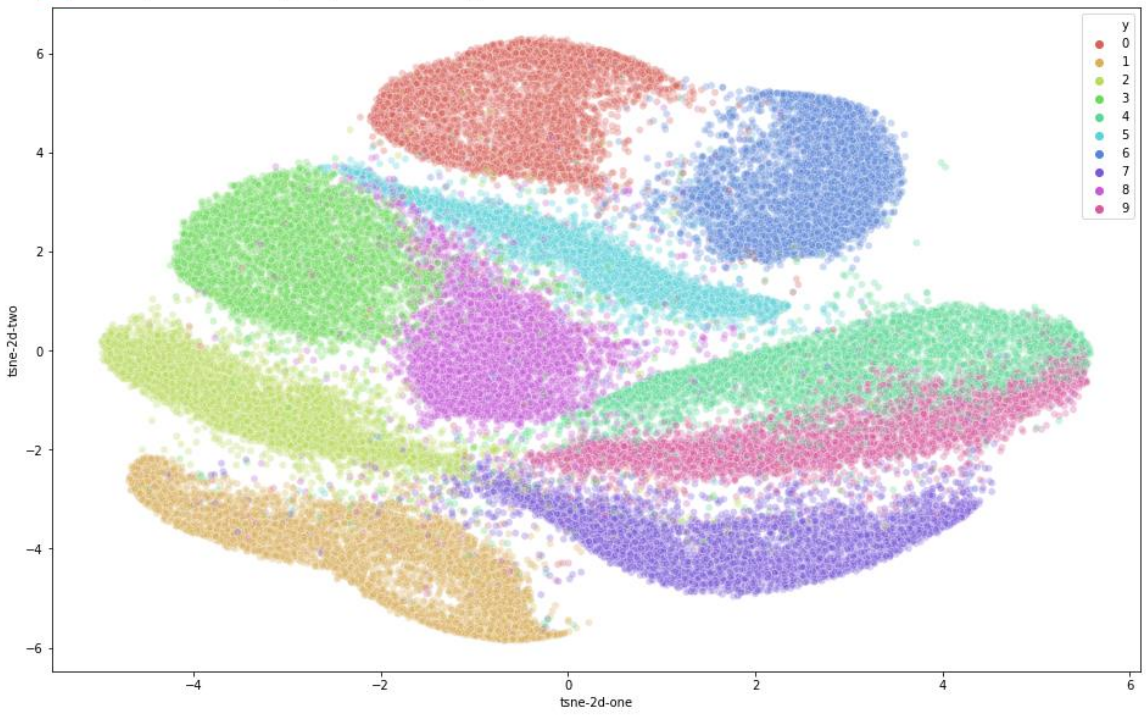
A slight different view of the data—



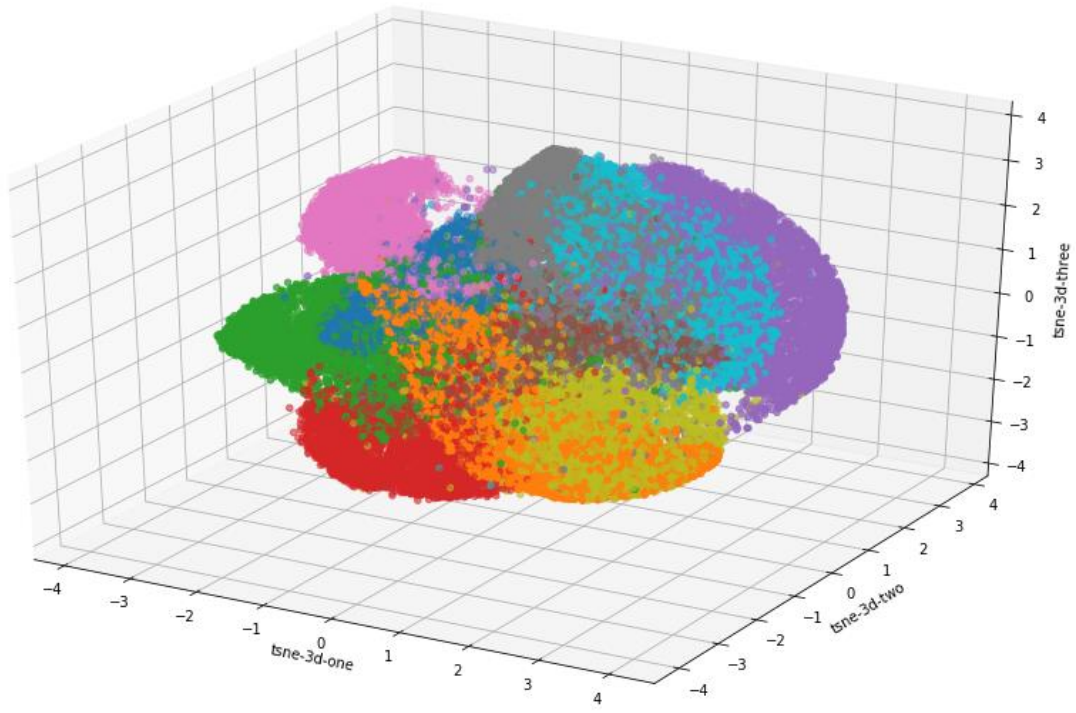


1(c) Using t-SNE to reduce the 'dataset1' to 2 dimensions and visualizing the scatter plot
[t-SNE done! Time elapsed: 2427.304451942444 seconds]
Observation: Each different color represents a different digit.

```
Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x2b843a12278>
```



1(d) Reducing the original 'dataset1' to the three dimensions and visualizing the scatter plot again
[t-SNE done! Time elapsed: 7577.2545647621155 seconds**]**
Observation: Each cluster represents instances of single class.
>any important distinction in inference as compared to (c):: All clusters are more visible and distinctive in the plot of 1(c) as compared to the plot of 1(d).



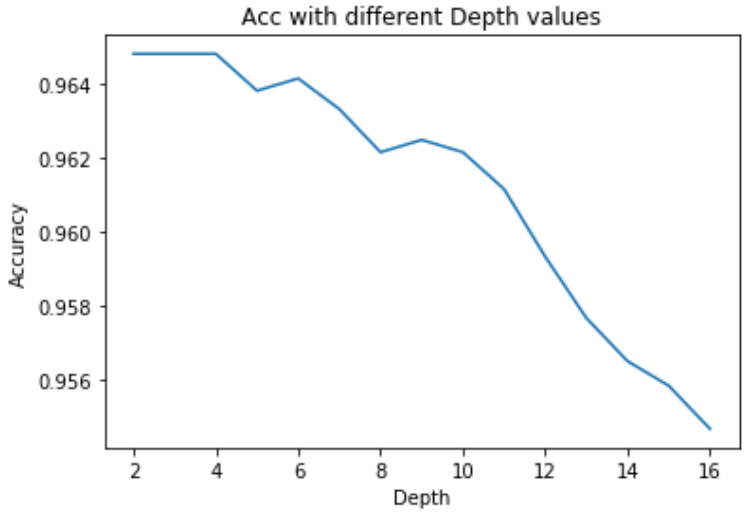
Details of Question 2

- Using Decision Tree Classifier
- Training Data size = 70% Testing Data Size = 30%
- Defined split_data(x,y,d_size,perc) function to split the data into 70:30 ratio. Also defined accuracy(pred, actual) function to calculate accuracy after model building on testing data.

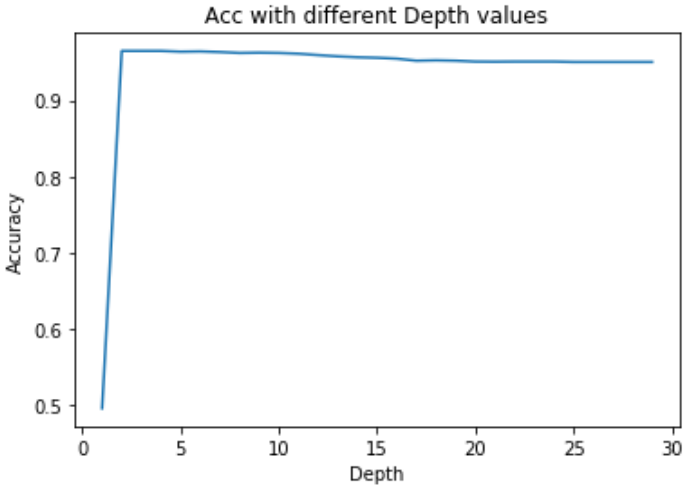
2(a) Taking depth as a hyperparameter, performed grid search for finding optimal value of depth.

>curve between depth and testing accuracy

With depth values[2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]



With depth values[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]



>the effect of depth on the performance of the classifier

For depth value =2 Accuracy is maximum(0.9648333333333333). Accuracy decreases slightly with increasing depth value.

>Is your best performance consistant with the visualization in (1.b).

Yes. As the best performance is at depth 2, which means it will have 4 leaf nodes for 4 classes present in the data. Each leaf node will be a range value.

2(b) Table representing train accuracy and validation accuracy for each value of the depth

Depth	Train accuracy	Validation_accuracy	Underfitting/Overfitting
2	0.9698333333333333,	0.9646666666666667,	None
3	0.9698333333333333,	0.9646666666666667,	None
4	0.97	0.9645,	Overfitting, as validation accuracy decreases after this
5	0.9706666666666667,	0.9633333333333334,	Overfitting
6	0.972,	0.9621666666666666,	Overfitting
7	0.9726666666666667,	0.9603333333333334,	Overfitting
8	0.974	0.9603333333333334,	Overfitting
9	0.9761666666666666,	0.9596666666666667,	Overfitting
10	0.9776666666666667	0.9588333333333333,	Overfitting
11	0.9796666666666667,	0.9565,	Overfitting
12	0.9813333333333333,	0.9568333333333333,	Overfitting
13	0.9835,	0.9556666666666667,	Overfitting
14	0.9855,	0.952,	Overfitting
15	0.9883333333333333,	0.9523333333333334,	Now testing accuracy is increasing slowly
16	0.99	0.9526666666666667	Now testing accuracy is increasing slowly

2(c) Table representing train accuracy and validation accuracy for each value of the depth using sklearn 'accuracy function'.

> Deviation between the results from implementation and inbuilt function:: Insignificant deviation as values upto two decimal points are exactly the same.

Depth	Train accuracy	Validation_accuracy
2	0.9677142857142857,	0.9646666666666667,
3	0.9677142857142857,	0.9646666666666667,
4	0.9677857142857142,	0.9645,
5	0.9684285714285714,	0.9633333333333334,
6	0.97,	0.9621666666666666,
7	0.9714285714285714,	0.9603333333333334,
8	0.9732142857142857,	0.9603333333333334,
9	0.9745,	0.9596666666666667,
10	0.9762857142857143,	0.9588333333333333,
11	0.978,	0.9565,
12	0.9807142857142858,	0.9568333333333333,
13	0.9823571428571428,	0.9556666666666667,
14	0.9855,	0.952,
15	0.9875,	0.9523333333333334,
16	0.9900714285714286	0.9526666666666667

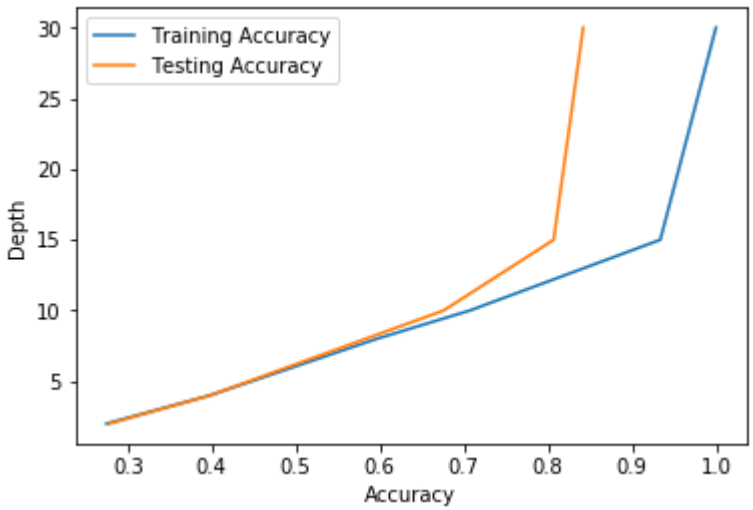
Details of Question 3

- Dataset : PRSA_data_2010.1.1-2014.12.31.csv
- Target Variable – month
- NA values were present in 'pm2.5'. Replaced those NA values using interpolate function.
- cbwd: Combined wind direction :: Values of this attribute were present in string format, changed them to integer values as follows:
  data.cbwd[data.cbwd == 'NW'] = 1
  data.cbwd[data.cbwd == 'SE'] = 2
  data.cbwd[data.cbwd == 'NE'] = 3
  data.cbwd[data.cbwd == 'cv'] = 4
- created following functions::
  split_data(x,y,d_size,perc) → to split the data by given **perc** ratio into training and testing sets.
  split_data_train(x,y,d_size)→to split the training data into 50% randomly for 100 stumps.
  ensemble(x_train, y_train, x_test, y_test, size, depth, stump)→for training 100 stumps
  maj_vot(size, clf, x_test, stump)→to find the class of an instance using majority voting concept
- 80:20 ratio of training and testing data

3(a)
Accuracy(for GINI) → 0.8350068461889548
Accuracy(for Entropy)→ 0.8450479233226837

Observation: Entropy gives better results. So in further questions, I have used entropy to choose between attributes.

3(b) Trained decision trees with different maximum depths [2, 4, 8, 10, 15, 30]



Best value of depth – 30
Accuracy for training for depth [2, 4, 8, 10, 15, 30]
[0.27477323292829026,
0.39799760397056305,
0.5966113297963375,
0.7072565462947116,
0.9337954247247419,

1.0]

Accuracy for testing

[0.278297581013236,
0.3982199908717481,
0.583409402099498,
0.6760611592879964,
0.8069374714742127,
0.8420812414422638]

3(c)

Created 100 different decision stumps (max depth 3). For each stump, 50% of training data has been taken for learning.

The accuracy on testing data - 0.3596531264262894

>*How is the performance effected as compared to part (a) and (b)::* There is a large performance gap between ensembling method(100 stumps) and just taking one tree with different depth values. Here depth of the tree is affecting the performance.

3(d) Tuning the decision stumps by changing the max-depth **[4, 8, 10, 15, 20, 30]** and number of trees**[5, 10, 20]**

Stump = 5

Depth	Training Accuracy	Testing Accuracy
4	0.4194072848626601	0.42081241442263806
8	0.6266293961607576	0.6095390232770425
10	0.7534371612756005	0.7082382473756276
15	0.9466598208682754	0.8405979005020539
20	0.971133550116949	0.851665905979005
30	0.972417924070852	0.8677544500228206

Stump = 10

Depth	Training Accuracy	Testing Accuracy
4	0.40691405915742035	0.40871748060246466
8	0.6550956958270344	0.6287083523505249
10	0.7857896688439487	0.742925604746691
15	0.9675119084971049	0.8792788680967595
20	0.9903020622379417	0.8945686900958466
30	0.9910151458969166	0.8978776814240073

Stump = 20

Depth	Training Accuracy	Testing Accuracy
4	0.41025129068142274	0.4149931538110452
8	0.6713254799053024	0.6501597444089456
10	0.7996520151744203	0.7550205385668645
15	0.9798625174705496	0.8943404837973528
20	0.9969480019395875	0.9128251939753537
30	0.9973473287886134	0.9147649475125513

Observation: Both training and testing accuracies increase as we increase the depth of the tree and the number of stumps. After depth=20, training accuracy does not significantly increase.

Final Observation for Q3:

SN	Testing Accuracy	Comments
(a)	0.8450479233226837	Entropy
(b)	0.8420812414422638	Max_depth = 30
(c)	0.3596531264262894	Max_depth = 3, Decision Trees ensembling = 100 stumps
(d)	0.9147649475125513	Max_depth = 30, Decision Trees ensembling = 20 stumps

Rank:: d > a=b > c