Lab -7

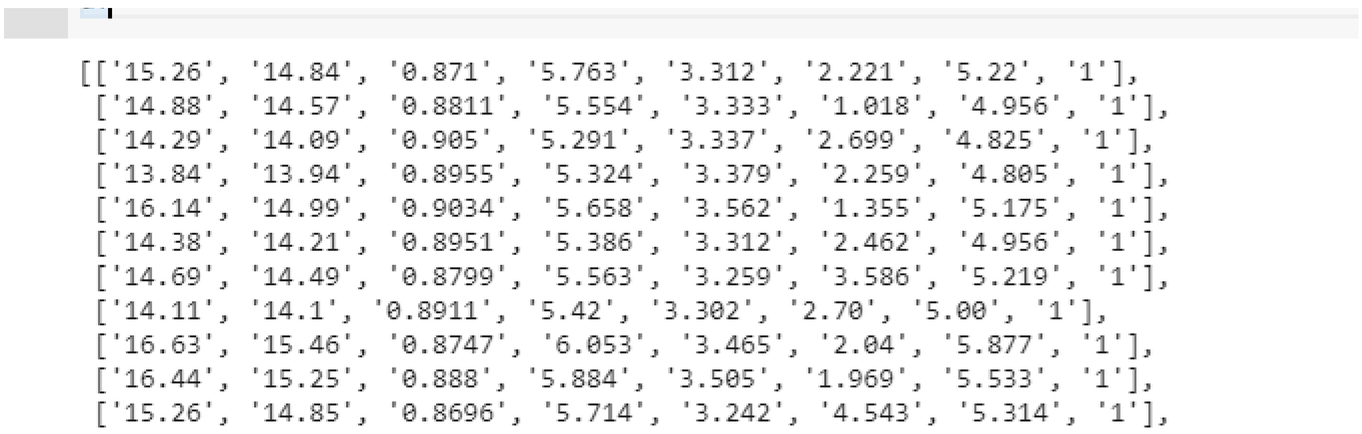
PRML
AY 2020-21 Trimester - III
Neural Network and Clustering

Name: Akshaykumar Kanani (B19EE008)

Q1: The objective of this assignment is to learn to implement Multi Layer Perceptron (MLP) from scratch using python. For this a nice tutorial has been provided in Resource-1. After implementing MLP from scratch, you need
to compare it with Sklearn's in-built implementation (resource-2). For this you
are supposed to use wheat seeds dataset provided in resource 1. (25 points)

Ans:

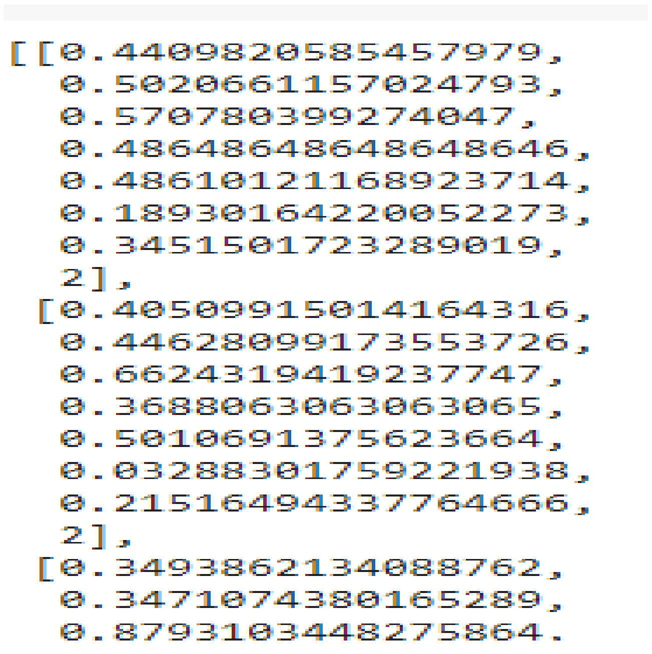
Given dats(just sample) is look like:



```
[['15.26', '14.84', '0.871', '5.763', '3.312', '2.221', '5.22', '1'],
 ['14.88', '14.57', '0.8811', '5.554', '3.333', '1.018', '4.956', '1'],
 ['14.29', '14.09', '0.905', '5.291', '3.337', '2.699', '4.825', '1'],
 ['13.84', '13.94', '0.8955', '5.324', '3.379', '2.259', '4.805', '1'],
 ['16.14', '14.99', '0.9034', '5.658', '3.562', '1.355', '5.175', '1'],
 ['14.38', '14.21', '0.8951', '5.386', '3.312', '2.462', '4.956', '1'],
 ['14.69', '14.49', '0.8799', '5.563', '3.259', '3.586', '5.219', '1'],
 ['14.11', '14.1', '0.8911', '5.42', '3.302', '2.70', '5.00', '1'],
 ['16.63', '15.46', '0.8747', '6.053', '3.465', '2.04', '5.877', '1'],
 ['16.44', '15.25', '0.888', '5.884', '3.505', '1.969', '5.533', '1'],
 ['15.26', '14.85', '0.8696', '5.714', '3.242', '4.543', '5.314', '1'],
```

Here last element of every sub array is our target element.
And by observation we can clearly see that the data is in string form so we first convert string in to float and int(for last colomn) and after that we normalized in between 0 and 1 as the data is wildly spread.

After this our data(just sample) is look like:



```
[[0.4409820585457979,
  0.5020661157024793,
  0.570780399274047,
  0.48648648648648646,
  0.48610121168923714,
  0.18930164220052273,
  0.3451501723289019,
  2],
 [0.40509915014164316,
  0.44628099173553726,
  0.6624319419237747,
  0.3688063063063065,
  0.5010691375623664,
  0.03288301759221938,
  0.21516494337764666,
  2],
 [0.3493862134088762,
  0.3471074380165289,
  0.8793103448275864.
```

And after preprocessing the data we come to know that every class hass equal number of element in it:

number of zero class member = 70
number of one class member = 70
number of two class member = 70

And for the algorithm we use some constant:

l_rate = learning rate= 0.3

n_epoch =number of epoch(cycle)= 500

n_folds = 5

n_hidden = 5

And after that we find following result:

Score: [95.23809523809523, 92.85714285714286, 97.61904761904762, 92.85714285714286, 90.47619047619048]

Mean Accuracy : 93.810%

And from the inbuild library we get the result:

Y prediction value:

```
[0. 1. 0. 2. 1. 1. 1. 2. 2. 2. 2. 0. 1. 1. 0. 2. 1. 1. 0. 1. 0. 1. 2. 0.
 0. 1. 2. 0. 1. 2. 2. 0. 2. 1. 0. 0. 0. 0. 0. 1. 2. 1. 0. 1. 0. 2. 0. 2.
 2. 1. 0. 0. 2. 2. 1. 1. 0. 1. 0. 1. 0. 2. 0.]
```

This value may change as we randomize the data in the code file.

And from this we got **accuracy of**: 96.82539682539682 %

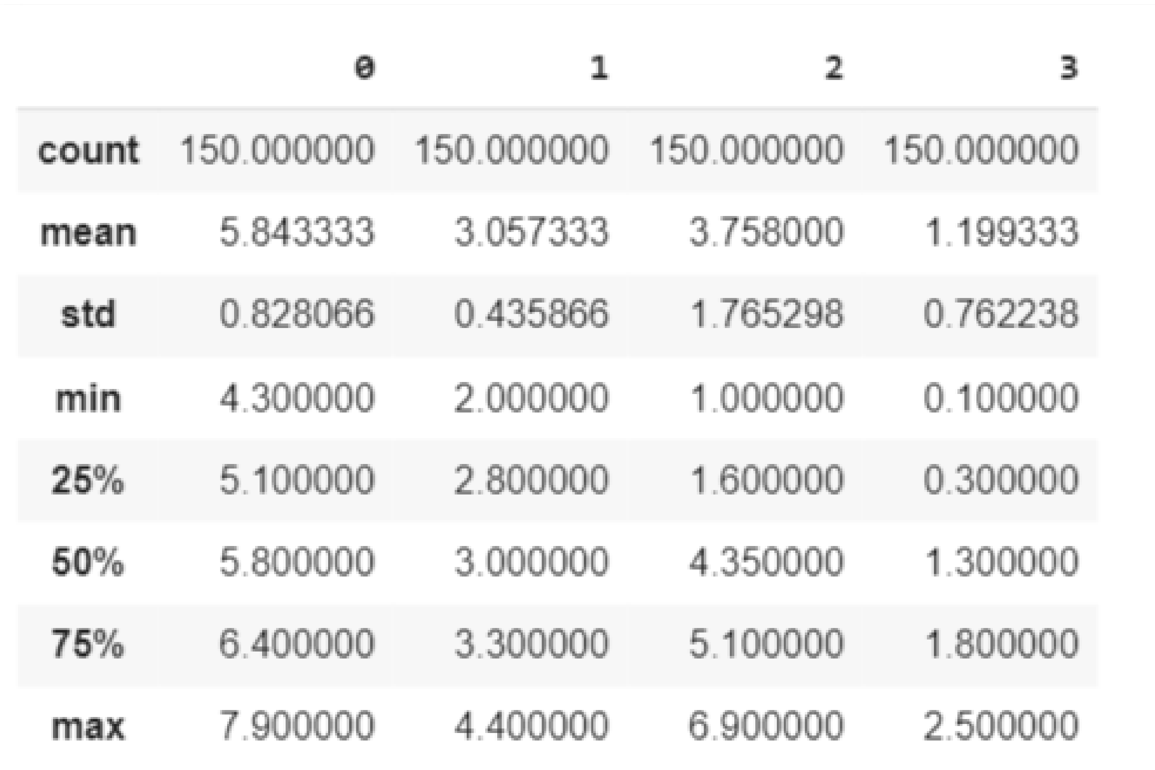
We got the difference in accuracy because of data randomization. But we can clearly see that the inbuilt method is very easy to use and also give more accurate answers.

Q2: You may use the MNIST dataset or any dataset for Face Images or Flower Images or Iris dataset for this Question.
Implement k-means clustering. Analyse the clusters formed for various values of k. Display the centroids of the clusters. DO NOT USE IN_BUILT ROUTINE for k-means clustering. (25 points)

Ans:

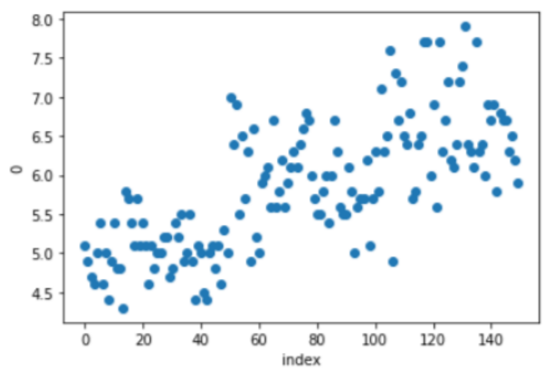
We have iris Dataset

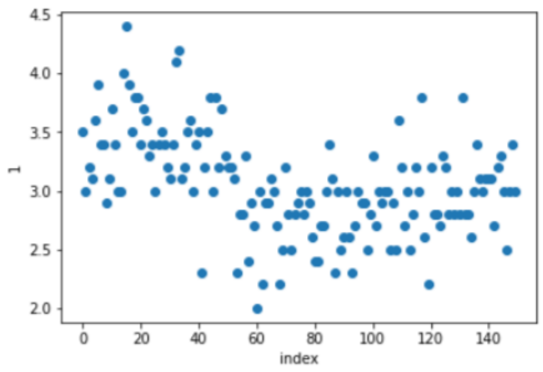
For input column we got:

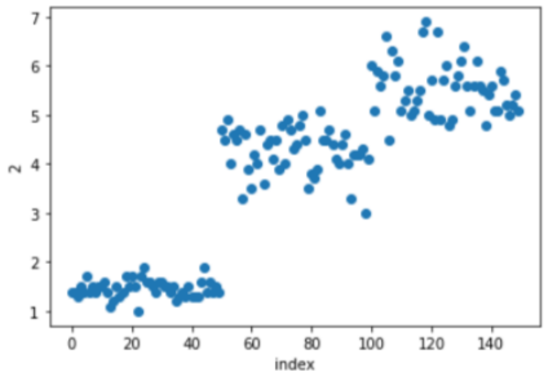


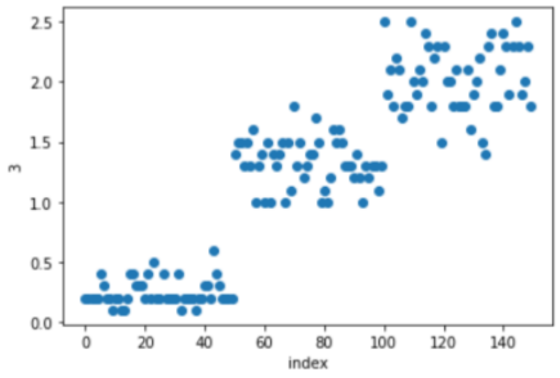
	0	1	2	3
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

And also we got the relation between index and other columns as follow:

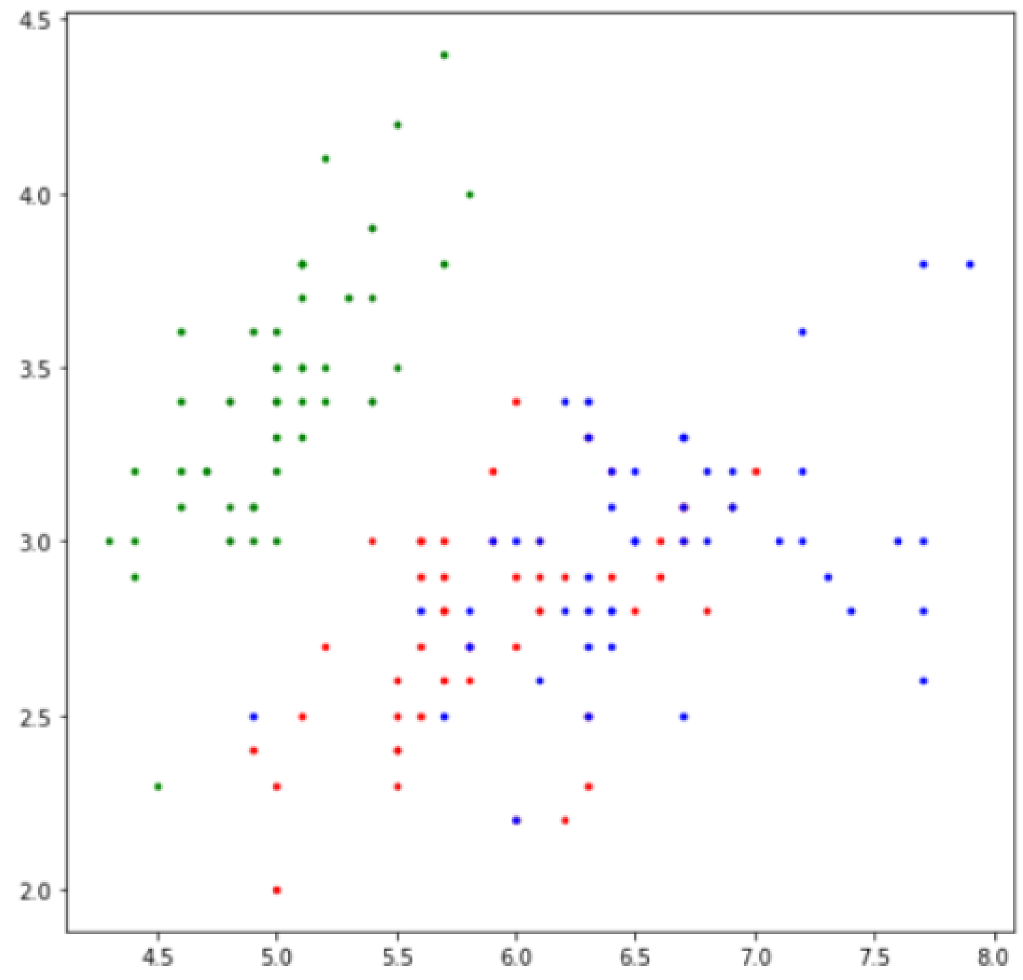








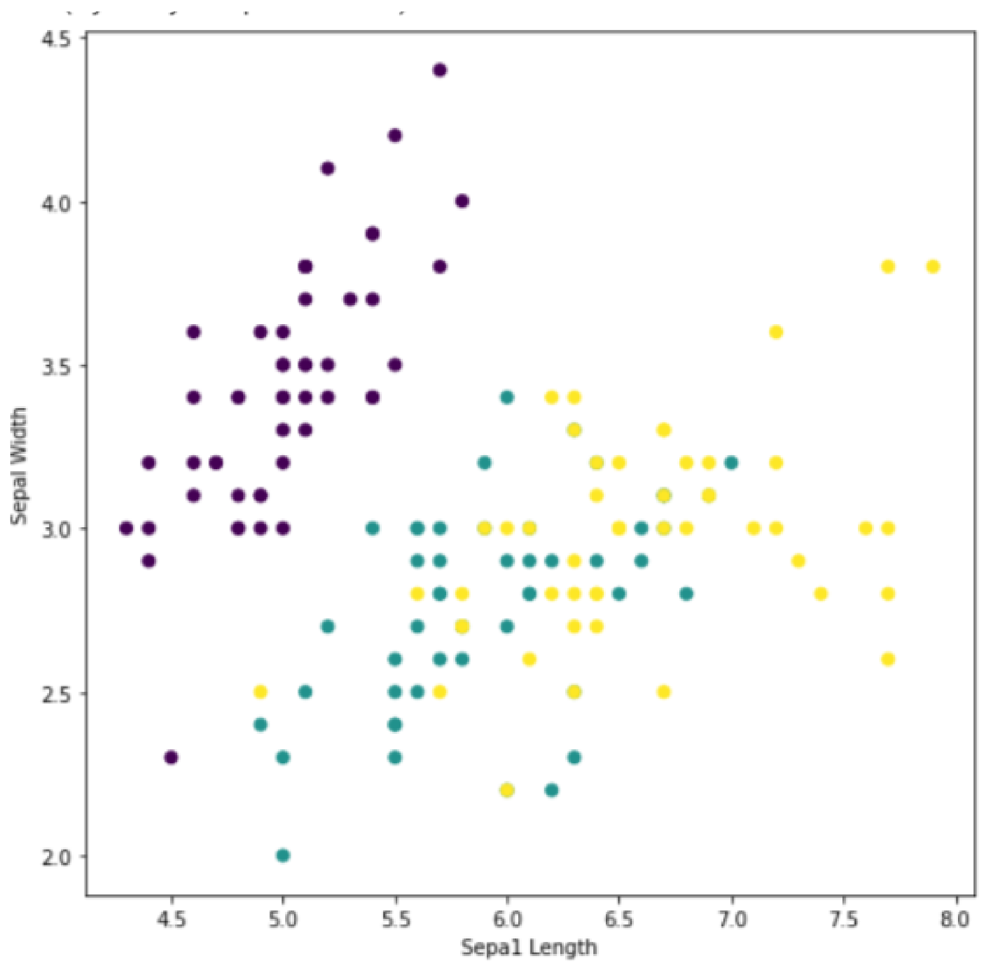
And for the first two colomn we plot the graph to find the relation and we got:



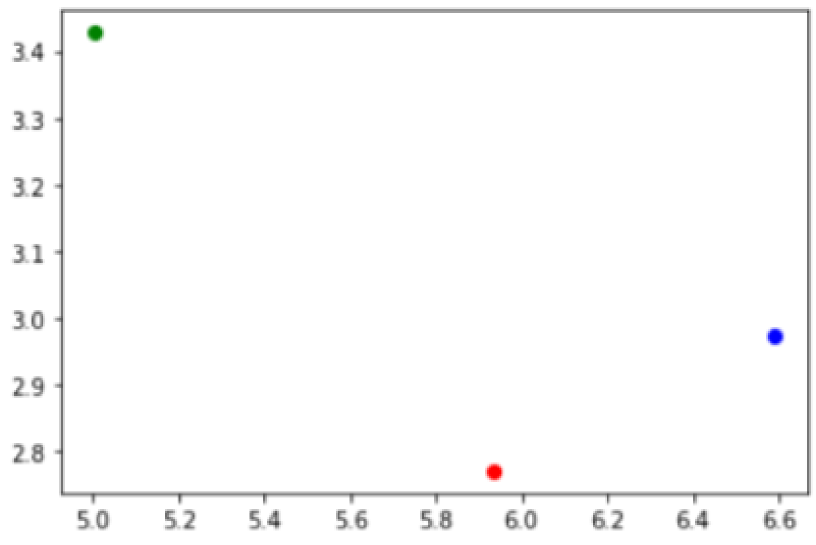
And after that we got to know that their are 3 target classes so we put k=3 in kmean algorithm(made from scratch in code file)
And we got first pair of Centroids as:

```
{0: [5.005999999999999, 3.428000000000001],
 1: [5.936, 2.7700000000000005],
 2: [6.587999999999998, 2.9739999999999998]}
```

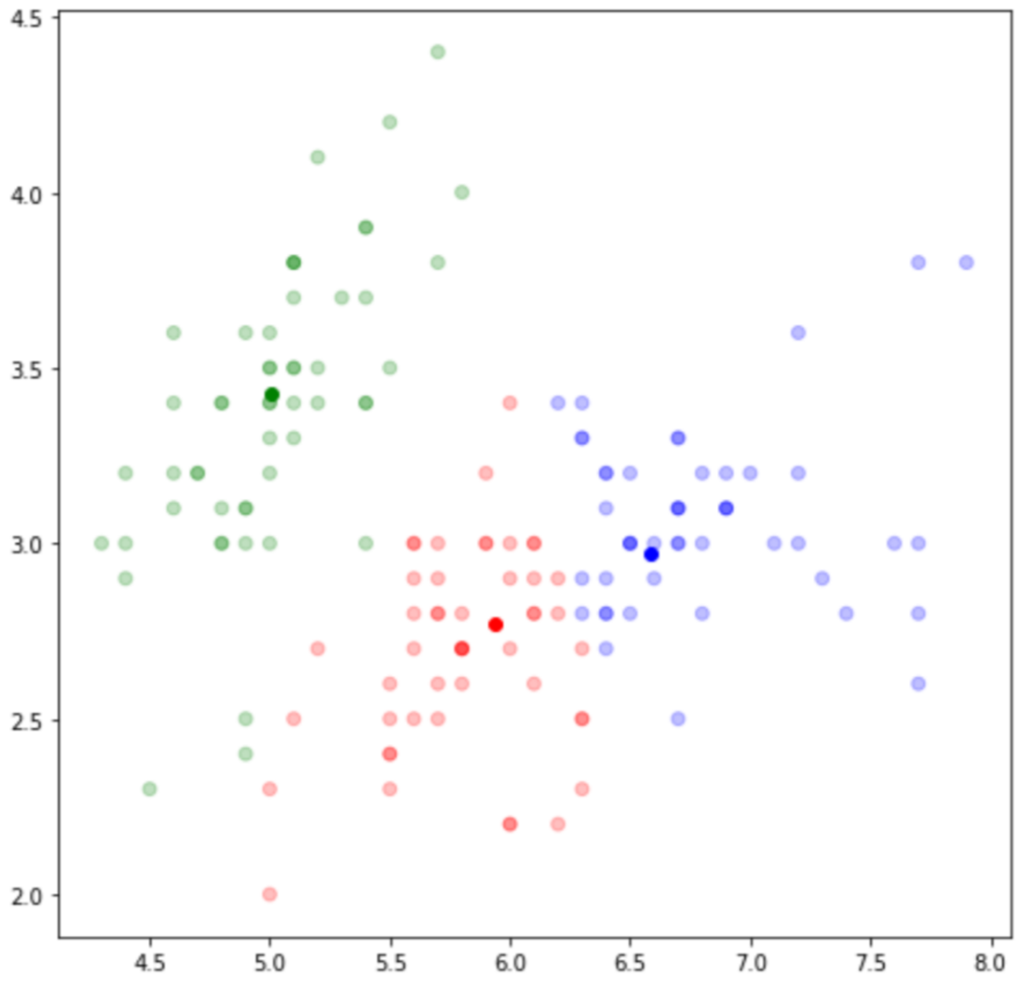
And the distribution as:



Centroid posion on plan:

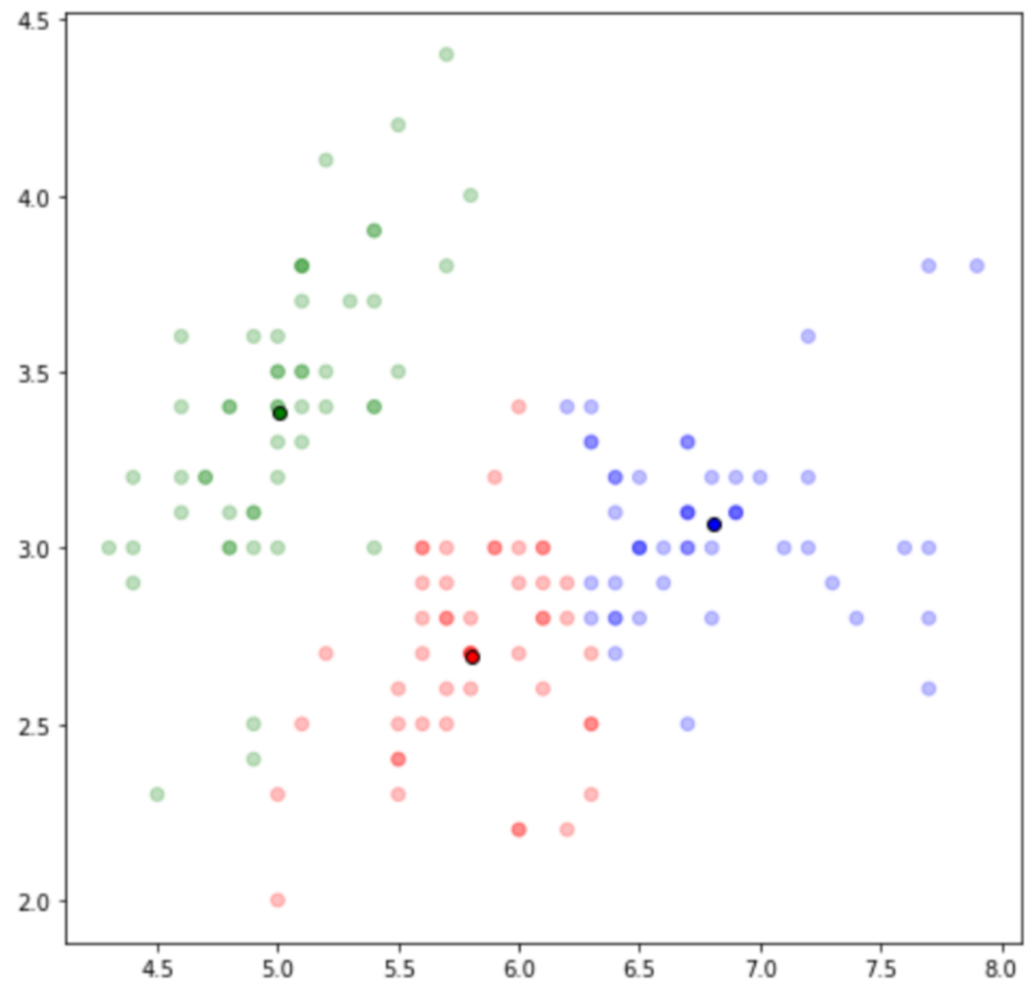


And after that we find the closest distance between centroids and all data points and add them to the dataset.And after classify them into different classes bayed on their distances we got :



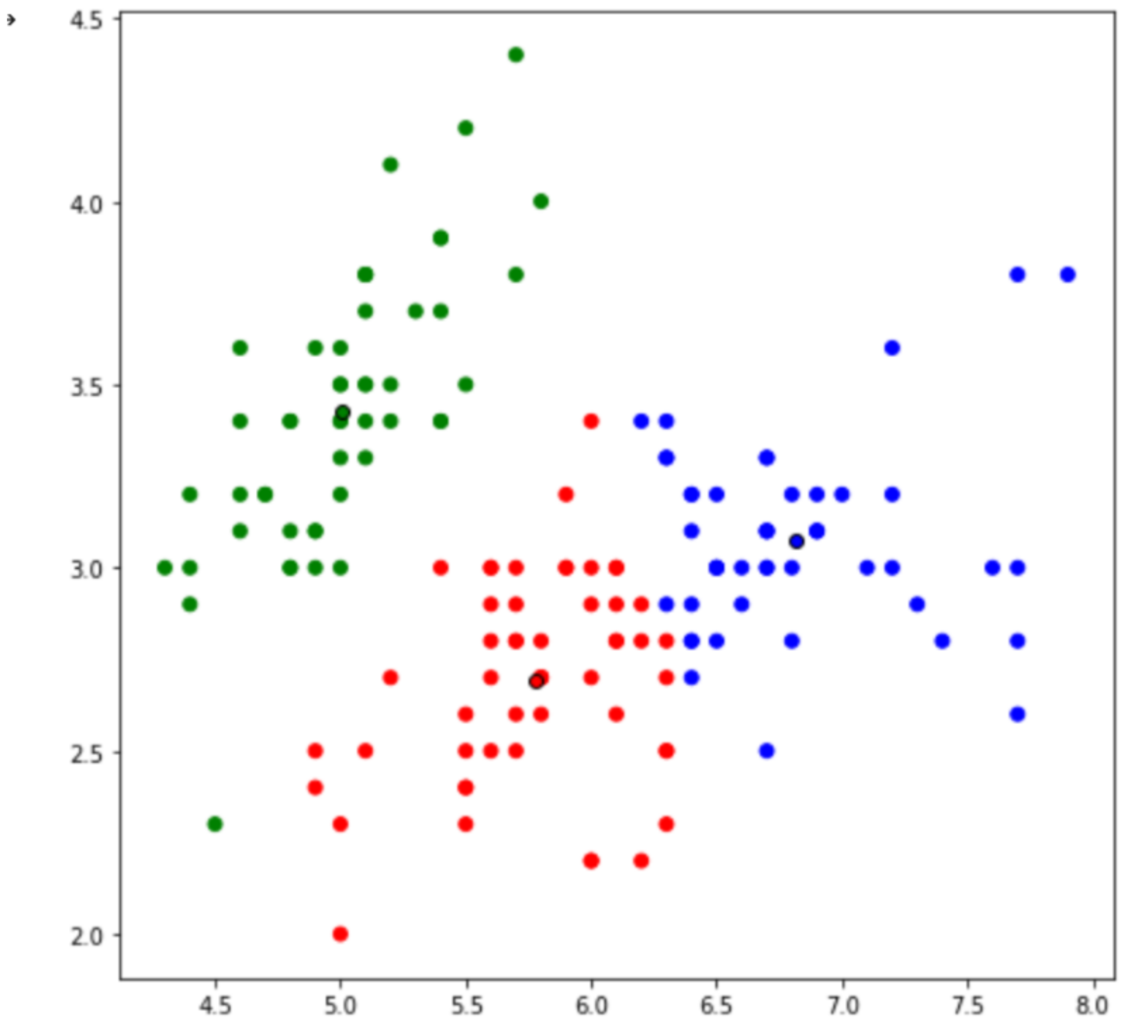
And after that we update the centroid based on the new data. And we got:
{0: [5.00943396226415, 3.383018867924529],
 1: [5.806122448979591, 2.693877551020408],
 2: [6.802083333333331, 3.0687499999999996]} as new centroids.

And again after classing we got:



And this way we find mean of dataset and do this whole process again and again.
And we repeat this process till our centroid become fixed(no change)
And so we got our final result as:
As final centroid point as:
{0: [5.005999999999999, 3.428000000000001],
 1: [5.773584905660377, 2.692452830188679],
 2: [6.812765957446806, 3.0744680851063824]} as centroids.

And clusters as follow:



Now all the data are separated in three clusters as their centroid after this is constant.

Thank you