CSE 6363 Machine Learning

Assignment 01

Abhijit Deshpande

UTA ID 1001677938

10/02/2020

K-MEANS

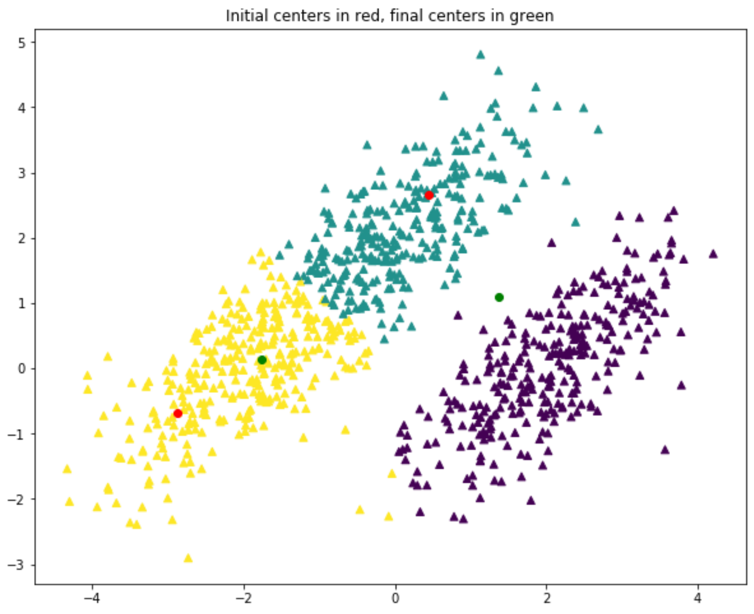
K-means clustering algorithm computes the centroids and iterates until it finds optimal centroid. It assumes that the number of clusters are already known. The number of clusters identified from data by algorithm is represented by 'K' in K-means. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early groupage is done. At this point we need to re-calculate k new centroids as barycenter of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop we may notice that the k centroids change their location step by step until no more changes are done. In other words, centroids do not move any more.

Problem 1 a

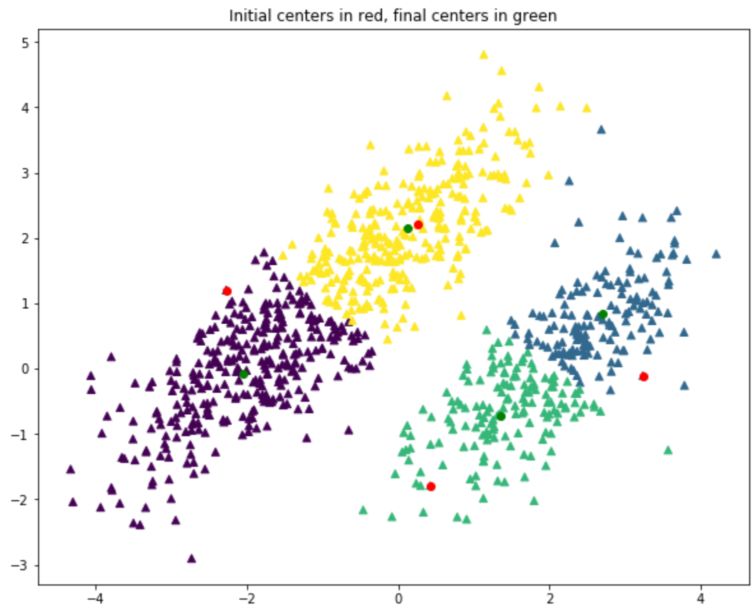
K-means is implemented using the data which is generated by the given parameters to create gaussian random samples. Clusters were generated in python using parameter np.random.multivariate_normal(mu,cov). The distance between two clusters has been calculated by Euclidean distance.For Euclidean distance np.argmin(np.sqrt(np.sum((self.centroids - x)**2, axis=1))) is used. myKmeans class is created to calculate centroids, this function helps to iteratively update the centroid using the Euclidean distance and fit the K clusters.

Using different K values (2,3,4,5) clusters are created and accuracy is calculated for respective K value. Initial Centroids are located by red and final centroids by green color. Accuracy is calculated by comparing initial class labels with predicted class labels. Accuracy are calculated for following K values:

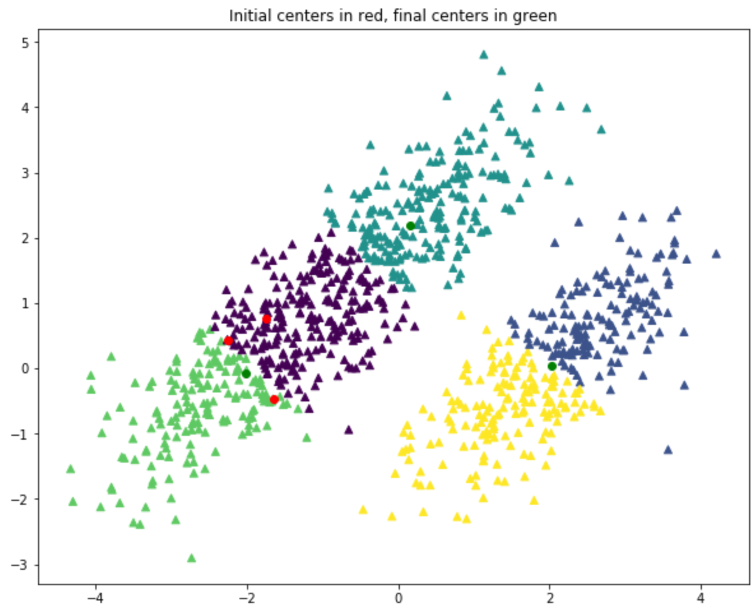
K=2 Accuracy for k=2 is: 0.03333333333333333



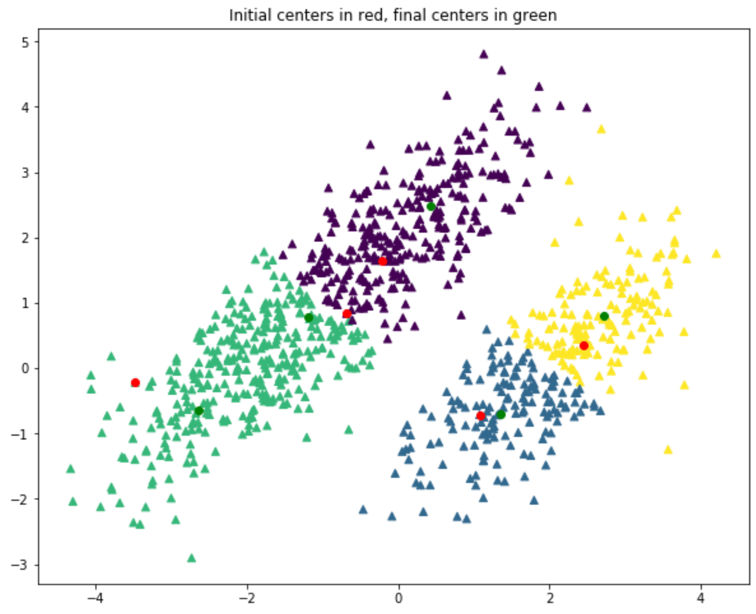
```
K=3 Accuracy for k=3 is: 0.05444444444444444
```



```
K=4 Accuracy for k=4 is: 0.2588888888888889
```



```
K=5 Accuracy for k=5 is: 0.5366666666666666
```



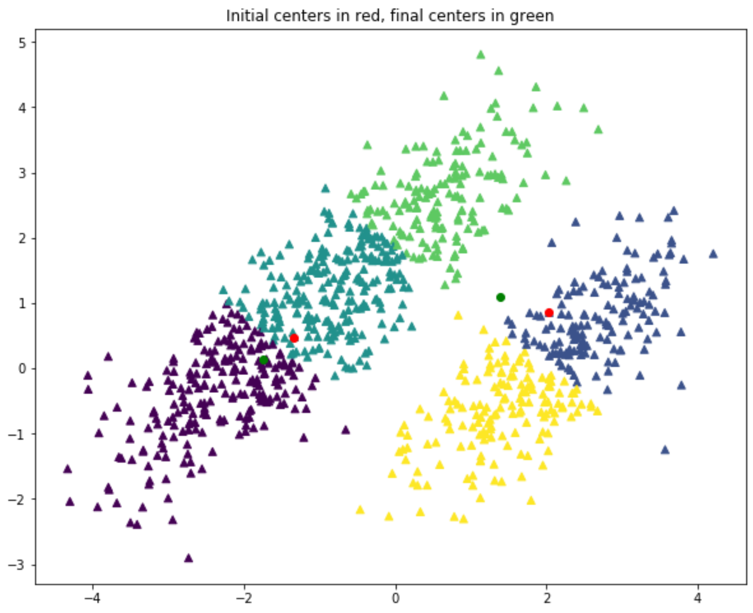
Model has highest accuracy with k=5 and have lowest value with k=2

Problem 1 c

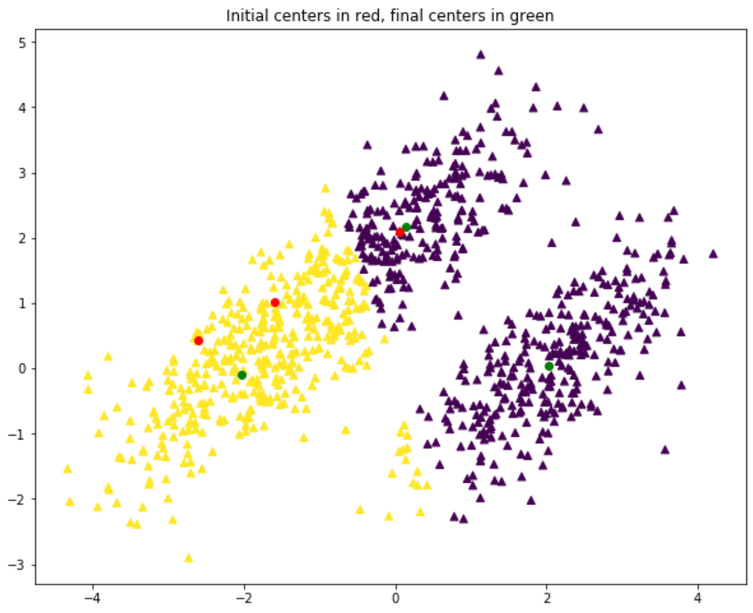
The same model is used with different input data and accuracies are checked for different clusters.

For k=2,3,4,5 the following accuracies were recorded:

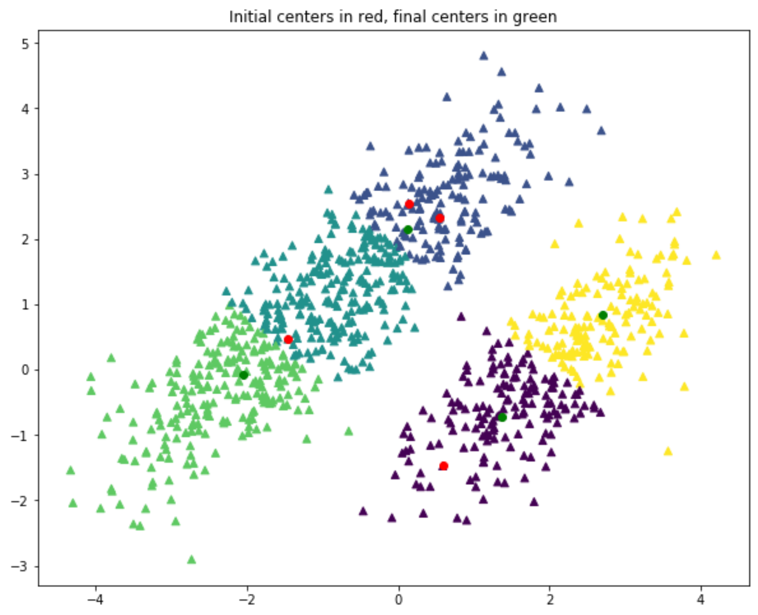
For k=2 Accuracy for k=2 is: 0.03222222222222222



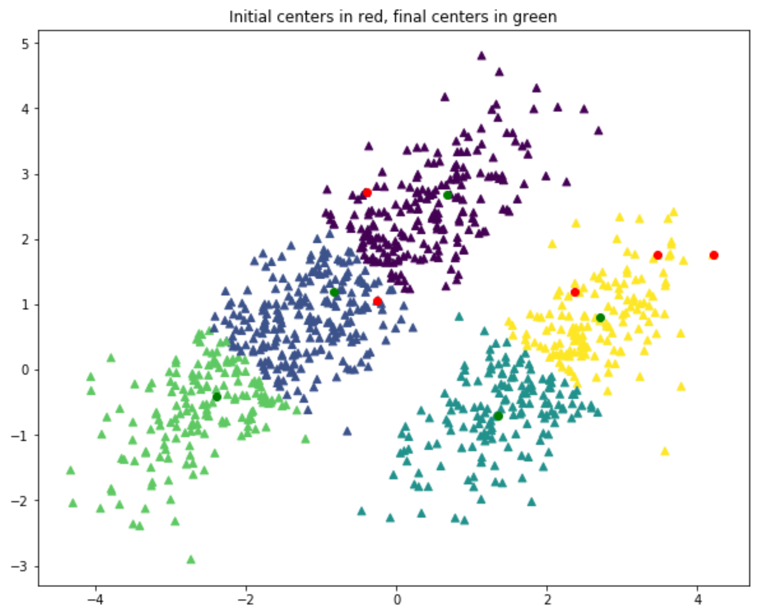
K=3 Accuracy for k=3 is: 0.29666666666666667



```
K=4 Accuracy for k=4 is: 0.4411111111111111
```



```
K=5 Accuracy for k=5 is: 0.15
```



```
Best accuracy found at k=4 with 0.411 and got low accuracy at k=2 with 0.032.
Model is performing better for greater k values.
```

Problem 2

K Nearest Neighbors:

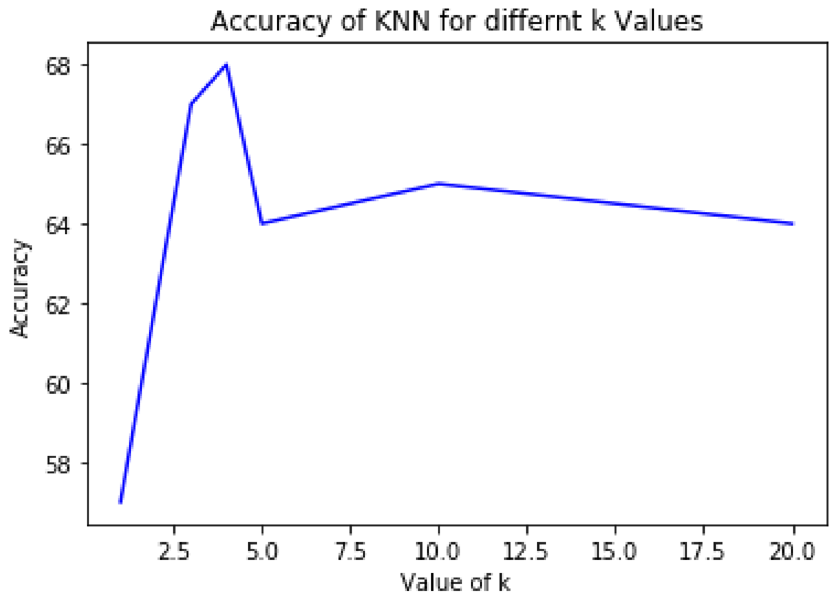
k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, normalizing the training data can improve its accuracy dramatically. Euclidean distance is most commonly used distance in for this algorithm. The best choice of k depends upon the data; generally, larger values of k reduces effect of the noise on the classification, but make boundaries between classes less distinct. A good k can be selected by various heuristic techniques. The special case where the class is predicted to be the class of the closest training sample (i.e. when k = 1) is called the nearest neighbor algorithm.

Problem 2 a

For KNN algorithm, the data is generated using given parameters. Gaussian random samples are generated using syntax np.random.multivariate_normal(mu,cov). The data is labeled with classes 0 and 1. 200 training points created and 50 used to test the model. KNN function created which predict the classes 0 and 1. Model is checked over different k values and accuracy are calculated over respective k value.

for k= [1,2,3,4,5,10,20] respective accuracies are

[56.999999999, 62.0, 67.0, 68.0, 64.0, 65.0, 64.0]



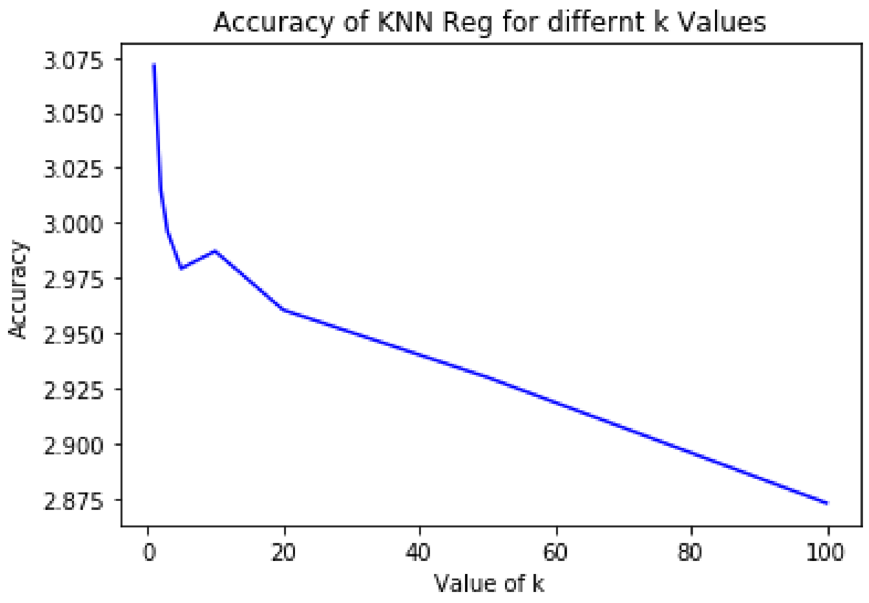
Problem 2 b

For KNN regression, regression function is used to predict the labels. Data is generated using gaussian random numbers and target variables assigned by y= 2x1+x2+ ε (ε = 0.5 is the noise). 300 training data points used to train the model and 100 test points used to test the accuracy and how well model perform. Using different k values the model is analyzed. Accuracy is calculated by root mean squared error (RMSE).

k= [1, 2, 3, 5, 10, 20, 50, 100]

accuracies with respective k values are following:

```
[3.0714272999655607, 3.0153585824743945, 2.9959795038049806, 2.9793331446575113, 2.987
2170432643808, 2.960571314001657, 2.9300649362298086, 2.872950247268717]
```



The low RSME value is 2.87 for k=100 and highest for k=1.

Problem 2 c

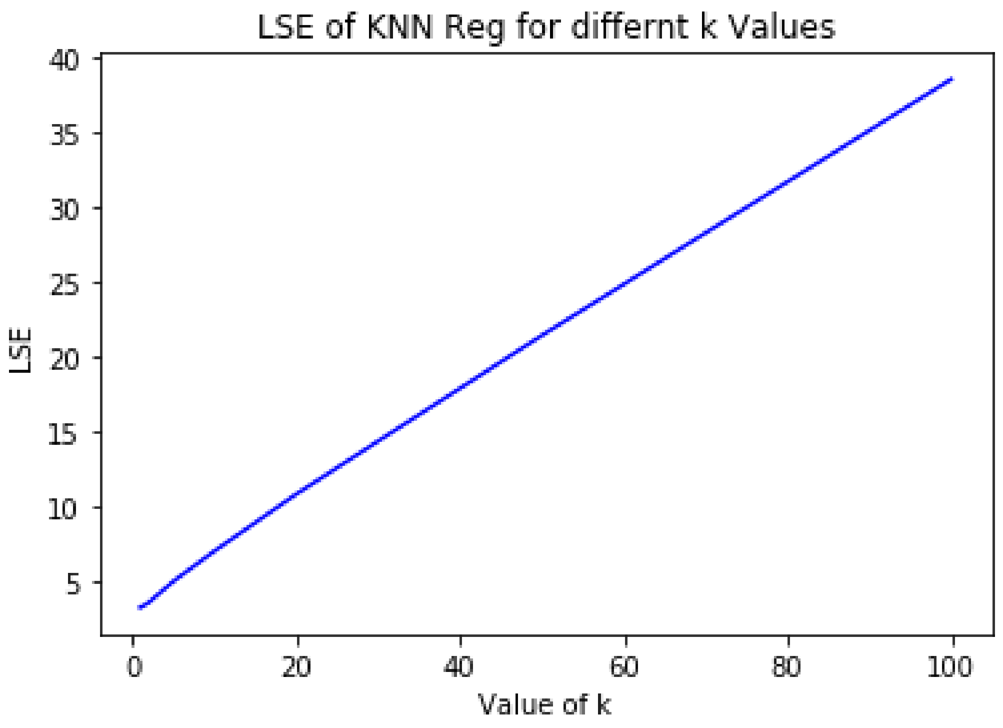
KNN locally weighted regression

Locally Weighted Nearest Neighbor algorithm (LWNN) uses the weighting scheme to re-scale the Euclidean distance between two data points when finding the nearest neighbor. Unlike the original KNN algorithm, the proposed LWNN algorithm has a training stage, which computes both the centroid and the associated independent weighting vector of each predefined class. Then, LWNN classifies a testing data point as the class of the centroid that has the minimum weighted distance to the examined point.

Model is analyzed on different k values and LSE is calculated for respective k value

k= [1, 2, 3, 5, 10, 20, 50, 100]

```
[3.240208932241926, 3.548892078601754, 4.029628418736429, 4.963495643105078, 6.979644
172347912, 10.798659301956455, 21.398190693335923, 38.5235322783018]
```



The low LSE value is 3.240208932241926 for k=1 and highest for k=100 with LSE=38.5235322783018

References

1. https://github.com/werberth/knn-implementation/blob/master/kNN%20Implementation%20using%20only%20Python/kNN%20Implementation%20using%20only%20Python.ipynb

2. https://towardsdatascience.com/build-knn-from-scratch-python-7b714c47631a