Monday, 1 st April 2019.

ASSIGNMENT2-CSC4007 ADCANCED MACHINE LEARNING REPORT

By Tianbai Peng 40120405

Summary

This assignment report talks about the whole process of using machine learning theory (mostly k-means clustering) to solve some real-world problems. Thorough the processing of solving the problem, it gradually describes how the functions used for implementation and the outputs received in each sub-task. And it's also demonstrated the machine learning theory used to solve every sub-task. And the previous knowledge point (cross validation and RBF feature) has been reviewed too. For example, what are the output got after executed, what is the meaning and interpretation of each result and the reason the results different from sub-task to sub-task and finally get the regular pattern combination with the data to find out the mysteries of machine learning.

1. Introduction

k-means clustering is a very important research field in machine learning and artificial intelligence and it is still kept very popular in nowadays, therefore it's very necessary to gain more knowledge and know how to implement it for this part. This assignment is very nice designed to let us know every kind of basic knowledge about k-means clustering. In this assignment, I mostly used k-means clustering to solve 3 real-world problem that classify the image data, fit an RBF model from data, and do the image segmentation. And this is what in the tasks:

- Using k-means clustering to cluster a dataset of hand-written letters.
- Using cross-validation to select hyper-parameters
- Applying k-means clustering for tuning the RBF features used in Ridge regression
- Applying k-means clustering for image segmentation

2. The details of each tasks

2.1 Task 1

In this task, I will use K-means clustering to cluster image data.

2.1.1 task 1.1

In this sub task I import the module of NumPy and some necessary module. I used ZipFile() function which is in zipfile module to unzip the data in file *"letters.zip*" and load all images to the 'letters' fold. Then I traversal all the images, for each image loaded, image be extracted as [:,:,0] then flatten it because we use only the "R" information for doing k-means clustering, finally the image data will be a matrix of 1800*16384 as below:

```
the shape of images: (1800, 16384)
```

Which clearly shows the image data contains 1800 images and the size of each image is 16384(128*128)

Then I set the seed as 2019 and shuffle the image data, I need to do this because the k-means clustering is sensitivity to the input data, K-Means algorithm selects the first K points as the center, the data input order is different, and the selected K center points are different, so the clustering results are different, so a random data may be better for the results. The root of this is the sensitivity of K-Means algorithm to the initial clustering center.

2.1.2 task 1.2

In this sub task, at the beginning I spilt the image data to train data (80%) and test data (20%) and then I used the find_clusters() function to do the k-means cluster for the both train data and test data. In this sub task I add a Euclidean_distance() function which used for computing the distance from all data point to all centroid, a computing_SSE() function which is used for computing the SSE error for clustering, these two functions has been used in the find_clusters() functions, and this is the main function to do the k-means clustering. The way to do k-means clustering has following steps:

1. Randomly choose clusters: pick a random data point to set as a centroid (try to pick clusters centroids, here we set random seed as 2019)
2. computing the distance from all data points to all centroids
3. Assign labels based on closest centroid.
4. Find/update new centroids from means of points
5. Check for convergence, if no new centroid then finished
6. Computing SSE and repeat 2-5

And I also add a MSE_computing() function here used for computing MSE error which will be used in the after tasks.

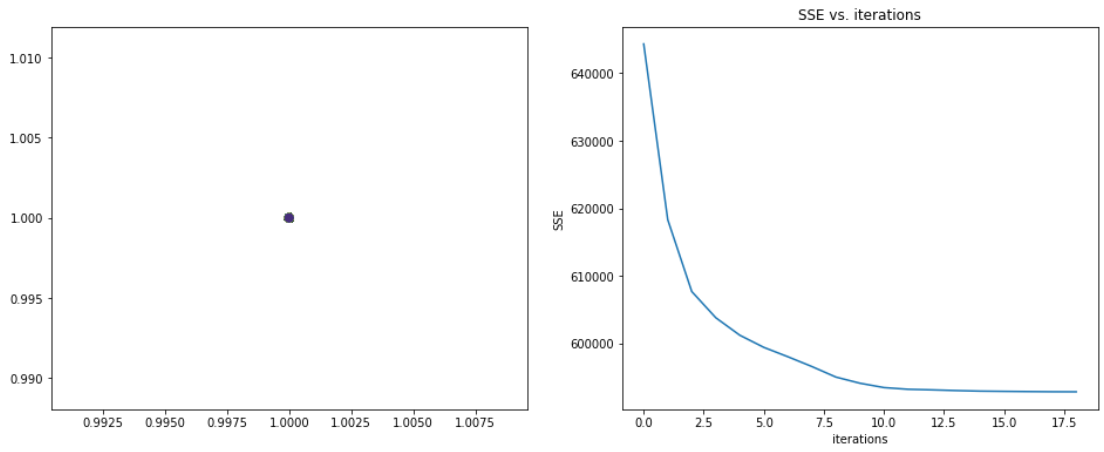
After clustering the train data and test data and drawing relative graphs and image. Finally, I got the cluster results for this task:

1) the SSE error for both the training and testing sets:

```
After 19 iteration SSE error for the training data is: 592804.30
69980715

After 16 iteration the SSE error for the testing data is 147693.
95955498077
```

2) plotting the SSE error during training vs. iterations of k-means:

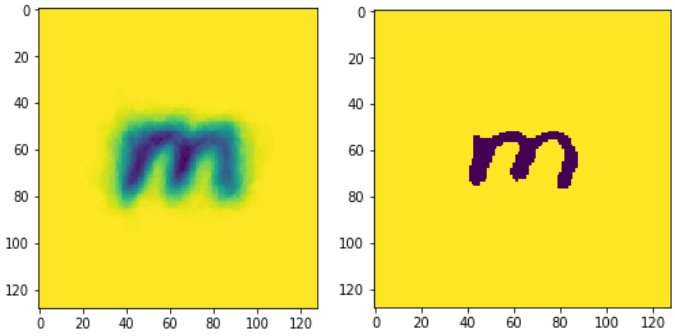


3) drawing the 9 image centroids
4) for each centroid, drawing its closest image in the training set.

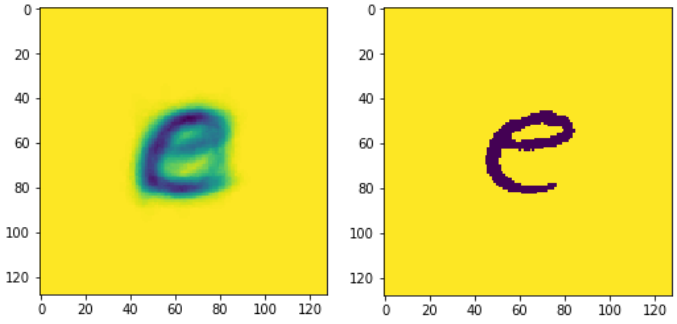
In this part ,my theory is traversal all the image data which stored in the assignment array , then calculate the distance between each image and centroid which stored in the centroids array, here the image I choose is in the same cluster with the centroid, finally the smallest distance image is the closest one for this centroid.

Outputs:

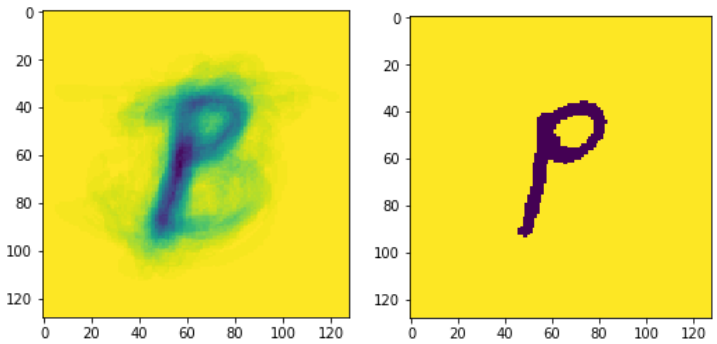
```
drawing the 9 image centroids and its closest image:
centroid 1 and the closest image to the centroid 1:
```



```
centroid 2 and the closest image to the centroid 2:
```

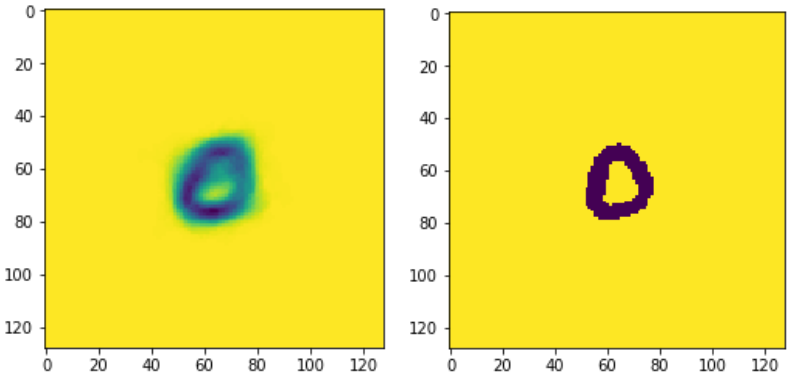


```
centroid 3 and the closest image to the centroid 3:
```



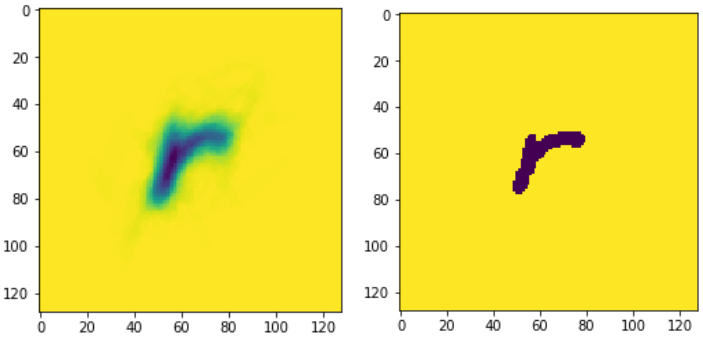
centroid 4 and the closest image to the centroid 4:

:

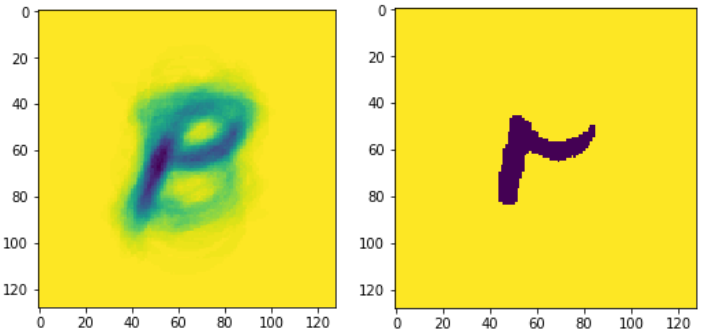


centroid 5 and the closest image to the centroid 5:

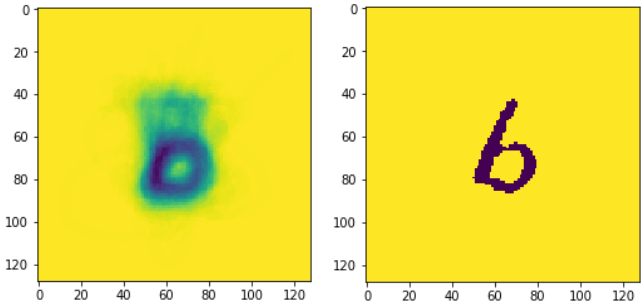
:



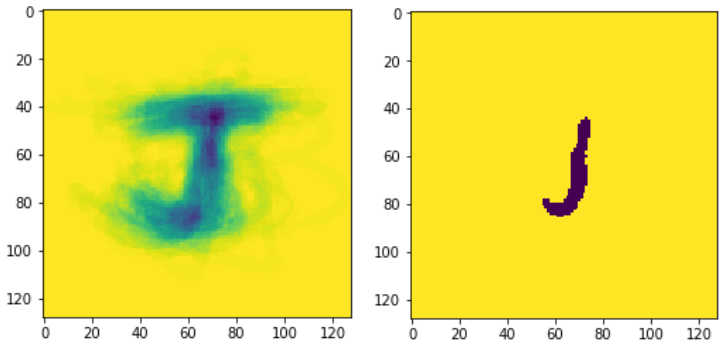
centroid 6 and the closest image to the centroid 6:



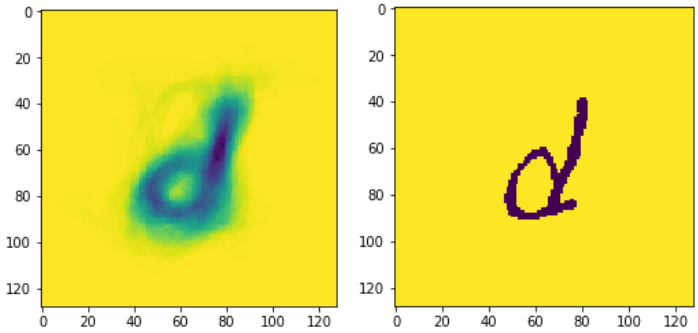
centroid 7 and the closest image to the centroid 7:



centroid 8 and the closest image to the centroid 8:



centroid 9 and the closest image to the centroid 9:



From the outputs, we can see that each centroid clearly shows one letter and the closest images are also looks like it's centroid. It prove that the k-means clustering algorithm works well and the accuracy is very high.

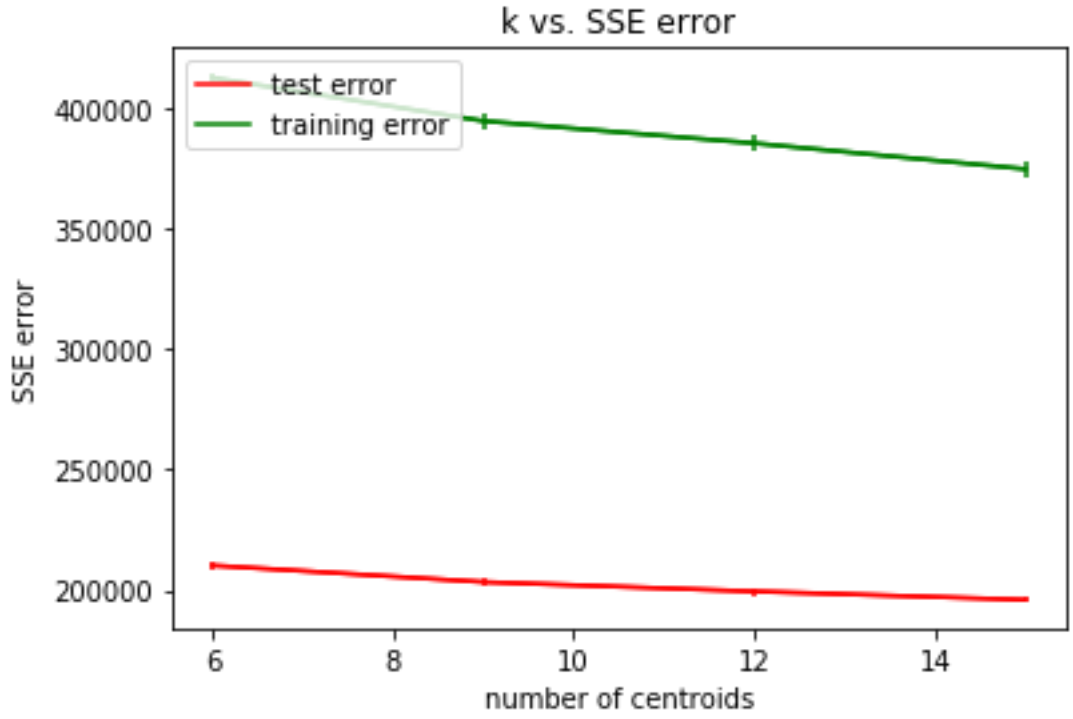
2.1.3 task 1.3

In this sub task, I will use cross-validation to select hyper-parameters .at the beginning I add a K_fold_cross_validation () function and this is the main function to do 3-fold cross validation. The way to do cross validation has following steps:

1. Validation data's size and loop over k fold from step 2 to step 5 (each time leave one out to make a test set)
2. Clustering on training
3. Computing error on the training dataset
4. Computing error on the testing dataset
5. Add this test/training error for computing statistics
6. Returning the mean and standard deviation of the cross-validated errors

Then I loop over num_candidates to find the best number of centroids, From centroids candidate{6,9,12,15} ,for each candidate, I pass it to the K_fold_cross_validation () and calculate it's SSE , and finally find the minimum SSE and corresponding number of centroids, the output as below:

```
best number of centroids =  15
best best_SSE =  195985.15685872376
```



From the outputs, we can see that 15 is the best number of centroids to do the k-means clustering. And the SSE error will decrease when more centroid used.

2.2 Task 2

This task is doing the k-means clustering with supervised learning. As the experiment results in assignment 1, the performance of ridge regression relies on the quality of features. So, in this task, I will use k-means clustering to create a set of radial basis function features (RBF) for ridge regression.

2.2.1 Task 2.1

In this sub task the propose is clustering the data and find the centroids then fit the RBF model by using these centroids, and finally got the MSE error on both the training and testing sets.

Firstly I import the module of NumPy and used loadtxt() function from NumPy to load data from "regression.data", and then spilt it to train data(80%) and test data(20%) . From the result, we can see the data shape of each dataset:

```
data.shape: (600, 3)
trainingSet.shape: (480, 3)
testSet.shape: (120, 3)
X.shape: (480, 2)
y.shape: (480,)
X_test: (120, 2)
y_test: (120,)
```

Which clearly shows the data has been spilt to trainingSet(480 rows) and testSet(120 rows)

Then I used the find_clusters() functions, and this is the main function to do the k-means clustering which is already defined at the task 1, and got 3 centroids results as below:

```
[[-3.04088354 -4.41084914]
 [-2.38402429  2.67138057]
 [ 3.30697143  4.24590428]]
```

Finally, I fit the RBF model by using these centroids. I used the linear regression fit an RBF model and with a ridge regression. In this task I used a RBF_Features() function which used for building a RBF feature.

the formula to calculate beta:
β = (((X^T * X)^-1)* X^T)* y
and:
φ(*x*)=[1,φ1(*x*) ,φ2(*x*)]

$$X=\begin{bmatrix} 1 & e^{(-\|x1-c1\|/2\sigma^2)} & e^{(-\|x1-c2\|/2\sigma^2)} & \varphi(x1) \\ 1 & e^{(-\|x2-c1\|/2\sigma^2)} & e^{(-\|x2-c2\|/2\sigma^2)} & \varphi(x2) \\ 1 & e^{(-\|x3-c1\|/2\sigma^2)} & e^{(-\|x3-c2\|/2\sigma^2)} & \varphi(x3) \end{bmatrix}$$

Then used the same way to calculate the error, finally I got the training MSE error and testing MSE error as below:

```
Training MSE of RBF feature is 0.011919314843797385
Testing MSE of RBF feature is 0.020573832732820804
```

From the result, we can see that both the training and testing MSE are very low, which verify that the fitted RBF model is correct and the k-means clustering algorithm is also take an important role to select those centers to improve its accuracy.

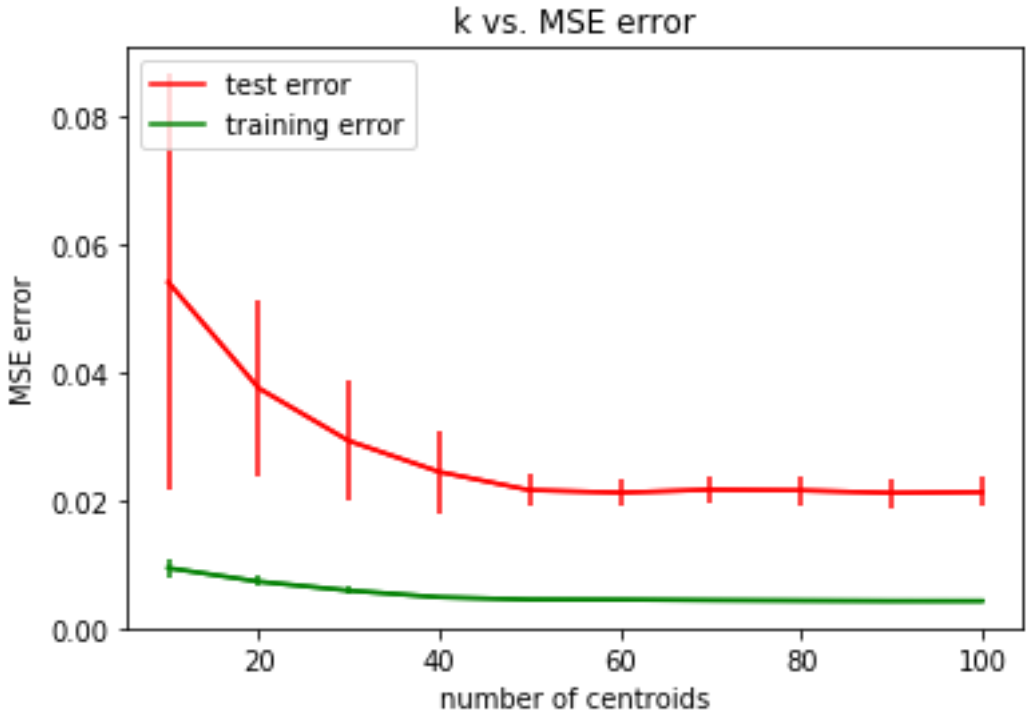
2.2.2 Task 2.2

In this task I used 5-fold-cross-validation select a better number of centroids *k* on the range $k=10*i$ where $i=1,2,\ldots,$ 10. It is similar to the task 1.3 but it contains all the information within task 2.1. Here I add a new five_fold_cross_validation_RBF() function to calculate the best number of centroids and the SSE and MSE error of the RBF models, inside the cross validation function I mostly using follow steps:

1.Validation data's size and loop over 5-fold from step 2 to step 5 (each time leave one out to make a test set)
2. separate the data into train set data and leave out data.
3.using k-means to find k centroids on train set data
4.create RBF features for train set using centroids as centers.
5.Add this test/training error(MSE , SSE) for computing statistics
6.Returning the mean and standard deviation of the cross-validated errors

And finally visualized the curves of training and validation error estimations (over 5 folds) together with their variances, the outputs as below:

1) the curves of training and validation mean square error estimations (over 5 folds) together with their variances:



2) the curves of training and validation SSE error estimations (over 5 folds) together with their variances:

Results:

```
best number of centroids =  100
best best_SSE =  73.73524175358287
```

so, from the result we can see that the best number of centroids is 100 which best SSE is 73.73

comparing to task 2.1 we used 3 centroids in that task and we got result

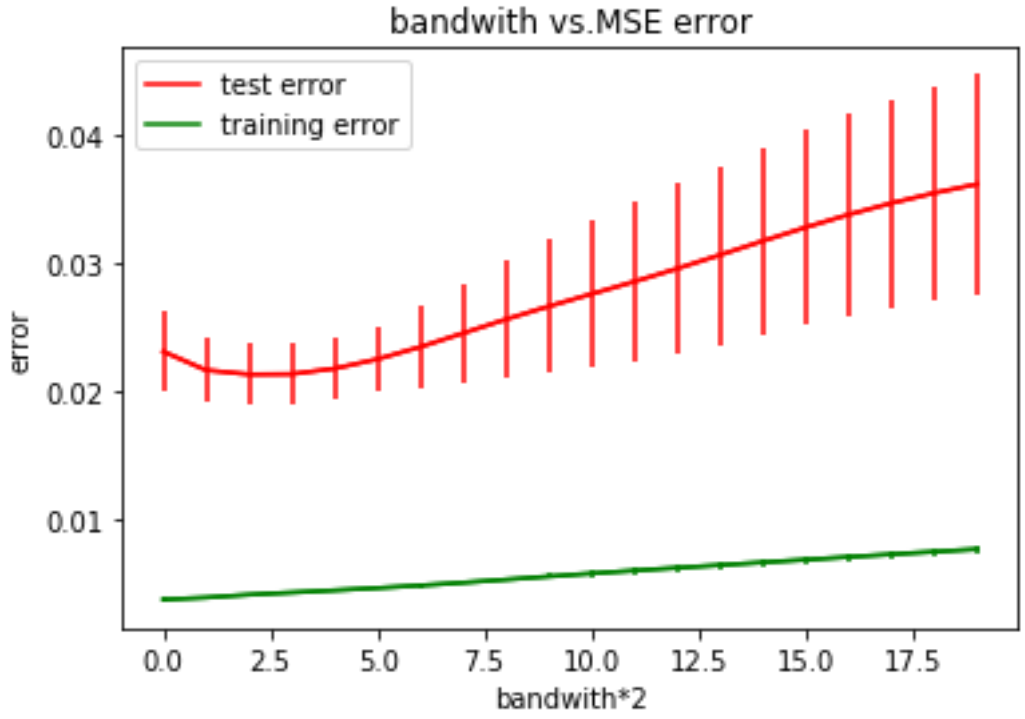
```
Training MSE of RBF feature is 0.011919314843797385
Testing MSE of RBF feature is 0.020573832732820804
```

And compare to the graph 'k vs MSE' above, this result is exactly the same as the chart shows. This indirectly proves here the correctness of k-means cl ustering and it also shows that the k-means clustering is an important role t o choose a correct hyper-parameter which would much better for the data p rocessing.

2.2.3 Task 2.3

The theory of this task is almost same as task 2.2 which is using the 5-fold-cross-validation, but the purpose of this task is to find out the best bandwidth for the RBF feature. Here I add a new five_fold_cross_validation_RBF2() function which is same as the one at task 2.2 but the different bandwidth as the argument is passed to the function to choose the best bandwidth, and finally got the output as below:

```
best bandwidth =  1.5
best best_cost =  0.021298557838071625
```



From the output we can see that when bandwidth is 1.5 , the MSE error is to a minimum .Therefore bandwidth = 1.5 is best for fitting a RBF model.

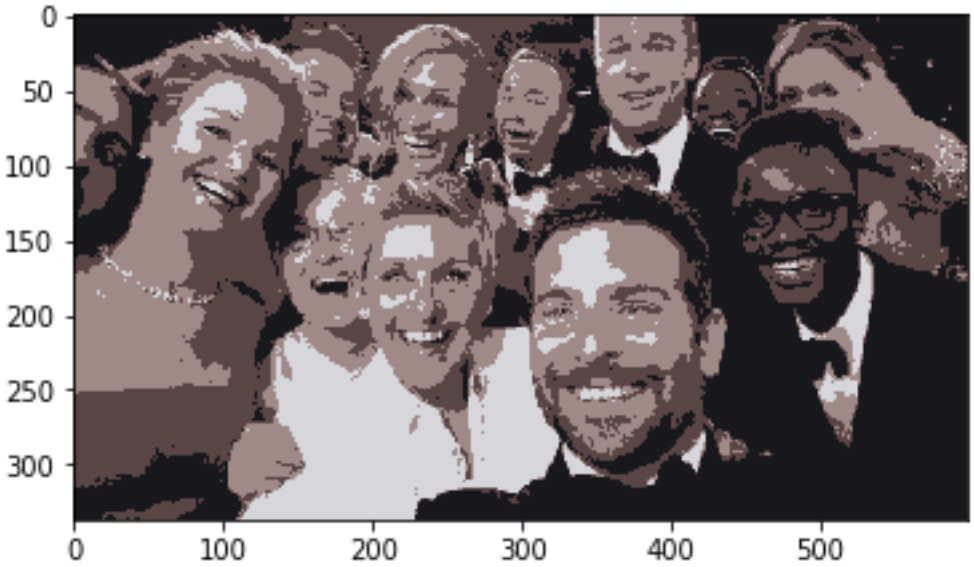
2.3 Task 3

The purpose of this sub task is segmenting a color image into several homogeneous regions. the K-Means clustering algorithm is used to group pixels to achieve image segmentation. In this task, image segmentation is realized based on color features and texture features.

Firstly, I load image ‘helen.jpg’, then I traversal this image, and try to get the information on each pixel and append those pixels information to one array, this make it much easier to do the K-Means clustering. After the clustering, we still need to assign pixel color to its centroid's color and finally the show the new image and 4 centroids as below:

```
centroids:
[[159.57668062 138.8515654  136.91679469]
 [214.93382753 214.06420405 219.03668803]
 [ 24.91488535  24.16925768  28.09901641]
 [ 90.34258222  69.41015847  70.35626274]]
```

New image with 4 clusters:



3. Conclusion

This is a very good assignment which helps us totally understand the basic knowledge about k-means clustering and basic thing about supervised learning and unsupervised learning. Thorough doing the assignment, I consider that there still be a lot of points need me to master, especially the cross-validation part and how to improve the accuracy and understanding every kind of error calculation more deeper. To this end, my next step is to learn other unknown information about machine learning for building a better AI programs in future.