Requirements:
Following are the steps in construction and use of the method:
a) Local Features: Compute SIFT features for each image.
b) **Reduce feature dimensionality** by performing principal component analysis (PCA) on the computed features. You may use any available OpenCV functions for this task or write your own. The possible useful functions are: **PCA**, **calcCovarMatrix** and **eigen.** Once the principal directions (eigenvectors corresponding to some number of largest eigenvalues) have been computed, the new feature vectors are determined by taking a dot product of these principal directions with the original feature vectors. It is recommended that you use about 20 components though this choice is left to you. We can call the new features as the PCA-SIFT features.
c) Codewords: Cluster the PCA-SIFT features from all the training images (regardless of their category) by using the k-means clustering algorithm. The means of the clusters define the *codewords* for vector quantization. Choice of the number of clusters may have a critical effect on the performance of the recognizer. You should test with a few values; it is recommended that you initially choose *k* to be around 100. For each feature in each image, find the codeword closest to it and assign it the code (or label) of that codeword.
There are some functions in OpenCV with "**BOW**" in their name; you may use these if you wish but writing your own code may be the easier path as the operations are quite simple.
d) Object Feature Vector: Compute a histogram of code words for each image; this defines the feature vector for the image.
e) Object Recognition: Compute the feature vector for each test image and use it to classify the category of that image. Note that the objects are not segmented from the background in this method; the entire image is classified as one. Use the *n*-nearest neighbor classifier. A suggested modification is that vote of each neighbor is weighted by its distance from the sample to be classified. The number of neighbors to be used is another parameter that you are encouraged to experiment with; we suggest that you set it to be no less than 10.
e) Error Analysis: For each test image, verify whether the given answer is correct or not. Show the detection rates for images in each category. Optionally, you may also want to compute a "confusion matrix".
f) Experimental Evaluation: You are encouraged to experiment with a number of parameters for each module (*e.g.* number of principal components, number of clusters and number of neighbors used in classification). The number of trials may be limited due to computational limits.

1. A brief description of the programs you write, including the source listing
Three functions:
plot the confusion matrix
http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html
Get Vector for train image

```
# Get Vector for test image
# Suggested Paramenters in the assginment descrpition
# Data Location
# Class and its labels in two format
# Initial the sift class
# the descriptors and filenames for Train Samples and Test Samples
# Compute SIFT descriptors and store image names
# convert Train and Test descriptor to numpy array
# Get PCA, mean and eigenvector
# project Train and Test features
# convert from numpy array to list
# kmeans algorithm
# get the histogram for each training image
# initite the knn class
# Test Iamge: histogram,label, accuracy
# Final Result
# Confusion matrix
# print the confusion matrix
# Plot non-normalized confusion matrix
# Plot normalized confusion matrix
```

2. Step by step results for one example to illustrate your method.
Number of components: 20 Number of clusters: 100 Number of neighbors: 10

```
********************************************************************************
Filename: Motorbikes/test/image_0109.jpg
Ground truth: Motorbikes
Result:       Motorbikes
Histogram:
[[ 4.  2.  4.  2.  3.  4.  4.  4.  6.  1.  5.  7.  0.  7.  3.  1.  1.  2.
   7.  3.  5.  3.  3.  3.  2.  1.  1.  4.  1.  3.  3.  1.  0.  2.  5.  2.
   4.  1.  3.  1.  1.  2.  1.  1.  4.  2.  1.  6.  0.  1.  3.  2.  4.  2.
   1.  3.  0.  3.  3.  0.  1.  6.  2.  2.  2.  1.  0.  1.  2.  4.  1.  2.
   2.  1.  9.  1.  3.  3.  2.  2.  2.  1.  2.  1.  0.  3.  3.  1.  0.  7.
   3.  0.  4.  1.  2.  2.  1.  2.  2.  2.]]
********************************************************************************
Filename: Motorbikes/test/image_0110.jpg
Ground truth: Motorbikes
Result:       Motorbikes
Histogram:
[[  3.   2.   0.   3.   3.   1.   5.   3.   4.   2.   1.   6.   1.   9.
    3.   6.   3.   5.   2.   1.   7.   4.   2.   3.   3.   2.   0.   2.
    3.   5.   1.   1.   0.   0.   4.   1.   1.   1.   3.   0.   6.   2.
    2.   1.   4.   6.   3.   3.   9.   3.   3.   9.   3.   0.   4.   2.
    2.   4.   5.   1.   6.   4.   3.   4.   3.   5.   2.   0.   5.   6.
    4.   2.   5.   8.  11.   2.   1.   3.   0.   3.   3.   3.   4.   2.
    1.   3.   1.   2.   4.   4.   4.   6.   1.   0.   4.   4.   0.   0.
    0.   1.]]
Motorbikes detection rate: 0.4
********************************************************************************
```

```
********************************************************************************
Filename: airplanes/test/image_0154.jpg
Ground truth: airplanes
Result:       faces
Histogram:
[[  0.   3.   4.   0.   7.   1.   2.   2.   0.   1.   3.  23.  10.   1.
    6.   4.   2.   3.   1.   5.   5.   9.   1.   5.   4.   0.   4.   2.
   21.   3.   1.   1.   0.   5.  10.   4.   0.   4.   2.   2.   0.   4.
    4.   6.   7.   3.   2.   1.   4.   1.   1.  10.   5.   2.   3.  10.
    6.   1.   3.   6.   8.   4.  14.   3.   5.  11.   1.  11.   3.   5.
    4.   1.   3.   3.   1.   4.   5.   4.   1.  11.   6.   2.   2.   1.
    1.   2.   5.   1.   0.   1.   3.  10.   2.   2.  14.   5.   1.   4.
    4.   6.]]
********************************************************************************
Filename: airplanes/test/image_0155.jpg
Ground truth: airplanes
Result:       airplanes
Histogram:
[[  1.   1.   5.   5.   4.   1.   4.   4.   1.   1.   4.  13.  16.   6.
    4.   5.   4.   4.   4.   4.   4.   6.   2.   5.   6.   5.   2.   4.
    9.   1.   1.   2.   1.   6.   5.   3.   1.   3.   2.   2.   1.   3.
    5.   5.   5.   1.   4.   5.   2.   2.   2.  17.  11.   2.   5.   4.
   10.   3.   2.   2.  14.   4.  10.   4.   1.   4.   2.   2.   2.   5.
    3.   1.   4.   9.   4.   8.   3.   2.   7.  11.  15.   1.   4.   1.
    2.   5.   1.   4.   3.   4.   7.  14.   6.   2.  15.   3.   2.   0.
    1.   4.]]
********************************************************************************
Filename: airplanes/test/image_0156.jpg
Ground truth: airplanes
Result:       faces
Histogram:
[[  1.   2.   1.   0.   4.   3.   4.   6.   0.   4.   5.  22.   9.   1.
    0.   8.   2.   2.   3.   5.   2.   9.   0.   1.   5.   0.   4.   4.
   14.   3.   2.   2.   1.   1.   1.   1.   4.   5.   3.   2.   3.   1.
    1.   9.   8.   1.   5.   1.   1.   2.   2.   5.  15.   0.   2.   0.
    7.   2.   5.   6.   8.   0.  11.   6.   2.  10.   1.   4.   0.   6.
    4.   3.   2.   2.   2.   5.   3.   3.   4.   5.   7.   2.   5.   5.
    2.   8.  11.   2.   0.   2.   1.  19.   5.   1.  14.   2.   1.   1.
    0.   4.]]
********************************************************************************
```

```
********************************************************************************
Filename: airplanes/test/image_0157.jpg
Ground truth: airplanes
Result:       faces
Histogram:
[[  1.   1.   2.   3.   2.   1.   6.   2.   3.   5.   5.  19.  10.   3.
    2.   5.   1.   7.   3.   1.   6.   2.   0.   3.   1.   2.   1.   4.
   11.   2.   1.   0.   0.   2.   6.   3.   1.   3.   3.   1.   1.   3.
    3.   2.   3.   3.   3.   2.   2.   0.   0.   2.   8.   2.   7.   5.
   10.   1.   2.   3.   9.   2.  11.   5.   3.   4.   1.   1.   3.   3.
    5.   5.   3.   2.   1.  10.   1.   2.   1.   8.   8.   1.   5.   2.
    1.   1.   0.   1.   2.   2.   3.  17.   2.   1.  13.   8.   1.   2.
    2.   2.]]
********************************************************************************
Filename: airplanes/test/image_0158.jpg
Ground truth: airplanes
Result:       Motorbikes
Histogram:
[[  2.   4.   4.   3.   0.   3.   7.   0.   2.   7.  10.  21.   9.   3.
    5.  21.   3.  10.   4.   1.   6.   4.   1.   6.   1.   5.   4.   7.
    8.  14.   0.   5.   1.   6.   4.   8.   2.   3.   1.   8.   2.   6.
   16.   5.   5.   3.   5.   1.   3.   1.   1.  16.  13.   3.   4.  13.
   30.   0.   5.   1.  19.   7.   7.   4.   1.  12.   1.   1.   5.   6.
    5.   6.   5.   2.   2.   3.   5.   1.   3.   1.   9.   2.   1.   2.
    3.   4.   8.   5.   0.   7.   6.   0.   4.   3.  10.   7.   2.   4.
    5.   9.]]
********************************************************************************
Filename: airplanes/test/image_0159.jpg
Ground truth: airplanes
Result:       airplanes
Histogram:
[[  1.   0.   1.   0.   3.   3.   3.   0.   1.   0.   1.   4.   3.   6.
    0.   3.   3.   2.   2.   3.   1.   1.   0.   0.   1.   0.   0.   5.
    2.   0.   6.   0.   1.   0.   2.   1.   0.   0.   4.   1.   5.   2.
    1.   0.   6.   3.   0.   2.   1.   3.   0.   3.   2.   3.   1.   0.
    5.   8.   3.   2.   0.   1.   2.   0.   5.   2.   2.   1.   1.   1.
    0.   3.   1.  10.   5.   0.   1.   0.   3.   3.  14.   1.   3.   2.
    3.   0.   2.   1.   1.   0.   1.   8.   4.   1.   1.   1.   4.   2.
    3.   0.]]
********************************************************************************
```

```
********************************************************************************
Filename: airplanes/test/image_0160.jpg
Ground truth: airplanes
Result:       airplanes
Histogram:
[[  1.   3.   5.   0.   1.   1.   9.   2.   0.   2.   2.  28.  17.   7.
    3.   7.   2.   0.   3.   3.   5.  19.   0.   1.   7.   1.   1.   2.
   10.   1.  10.   2.   1.   2.   0.  11.   0.   1.   6.   2.   3.   4.
    3.   7.   8.   1.   2.   1.   6.   4.   4.  13.  12.  11.   7.   1.
    6.   4.   2.   4.   9.   2.  11.   1.   1.   5.   3.   1.   2.   1.
    3.   4.   1.  16.   3.  10.   0.   4.   5.   7.  28.   2.   7.   3.
    2.   1.  17.   2.   3.   3.   2.   6.   9.   0.  11.   2.   2.   1.
    2.   1.]]
airplanes detection rate: 0.5
********************************************************************************
********************************************************************************
Filename: car_side/test/image_0021.jpg
Ground truth: car_side
Result:       car_side
Histogram:
[[  0.   2.   5.   4.   1.   8.   3.   3.   1.   4.   1.   8.   1.   4.
    5.   3.   9.   4.   6.   0.   3.   9.   5.   3.   3.   2.   2.   3.
    1.   0.   3.   2.   5.   2.   1.   8.   0.   6.   5.   9.   1.   6.
    3.   2.   1.   3.   2.   1.   1.   3.   4.   4.  20.   4.   0.  10.
    1.   4.   5.   1.   1.   5.   3.   3.   2.   4.   4.   1.   4.   5.
    1.   2.   8.   4.   3.   0.   5.   5.   1.   1.   5.   1.   3.   6.
    1.   2.   7.   2.   3.   3.  10.   5.   7.   3.   3.  13.   7.   8.
    6.   3.]]
********************************************************************************
Filename: car_side/test/image_0022.jpg
Ground truth: car_side
Result:       car_side
Histogram:
[[  2.   3.   8.  10.   2.  10.   3.   4.   4.  10.   2.   6.   2.   3.
    8.   5.   7.   6.   8.   2.   2.  16.   3.   2.   1.   3.   3.   9.
    3.   2.   6.   1.   9.   3.   3.  10.   2.   2.   9.   7.   2.   7.
    1.   2.   2.   5.   2.   1.   0.   5.   4.   5.  16.   4.   2.   8.
    2.   2.   5.   1.   0.   6.   2.   4.   3.   4.   4.   2.   8.   3.
    0.   6.   8.   1.   2.   3.   5.   3.   0.   1.   3.   4.   1.   6.
    0.   6.   7.   9.   1.   7.   9.   3.   9.   0.   4.  10.  11.   5.
   11.   4.]]
********************************************************************************
```

```
********************************************************************************
Filename: car_side/test/image_0023.jpg
Ground truth: car_side
Result:       car_side
Histogram:
[[  5.   6.  10.   5.   8.   4.   4.   3.   5.  10.   3.   5.   9.   3.
   13.   4.   9.   3.   4.   2.   5.   2.   5.   4.   6.   4.   8.   9.
    7.   4.   4.   9.   5.   9.   2.  14.   3.   6.   2.   8.   5.   5.
    5.  11.  11.   7.   7.   3.   3.  11.   6.   1.  15.   3.   4.   6.
    1.   4.   9.   2.   2.  15.   1.   3.  11.   4.   7.   5.  11.  11.
    4.   2.   8.   2.   3.   1.  12.   5.   2.   9.   3.   5.   4.  14.
    8.   7.   6.   7.   3.   8.   9.   4.   4.   8.   4.   5.   3.  10.
    8.   6.]]
********************************************************************************
Filename: car_side/test/image_0024.jpg
Ground truth: car_side
Result:       car_side
Histogram:
[[  3.   2.  10.   6.   2.   6.   6.   3.   5.   2.   9.   5.   7.  14.
    8.   5.   8.   3.   8.   4.  11.  10.   6.   5.   3.   2.  11.   7.
    2.   6.   2.  15.   8.  13.   4.  10.   5.   8.   4.  14.   3.   2.
    5.   3.   5.   3.   9.   3.   2.   5.   6.   3.  11.   2.   4.  11.
    3.   3.   4.   3.   4.  15.   2.   2.  11.   6.   9.   6.   5.   9.
    5.  11.   6.   5.   8.   4.   7.  14.   5.   6.   3.   4.   4.   8.
    2.   4.   6.   8.   9.   2.   6.   3.   5.  14.   1.   7.   8.   5.
    3.   6.]]
********************************************************************************
Filename: car_side/test/image_0025.jpg
Ground truth: car_side
Result:       car_side
Histogram:
[[  1.   7.   6.   6.   3.   8.   5.   0.   7.   5.   3.   4.  10.  11.
    8.   3.  10.   2.   2.   2.   6.   1.   4.   5.   3.  10.   7.   7.
    0.   4.   4.   7.   5.   2.   2.  12.   2.   3.   3.  10.   4.   2.
    6.   2.   8.   5.   8.   6.   2.  15.  11.   2.   1.   3.   1.  11.
    0.   6.   1.   0.   4.   6.   1.   6.   9.   3.  11.   0.   8.   5.
    6.   2.   5.   3.  12.   0.   4.   7.   5.   4.   2.   4.   4.   6.
    2.   0.   2.   3.   3.   4.   7.   4.   3.   4.   8.   9.   6.   4.
    3.   6.]]
********************************************************************************
```

```
********************************************************************************
Filename: car_side/test/image_0026.jpg
Ground truth: car_side
Result:       car_side
Histogram:
[[ 1.   8.   4.   5.   2.   3.   4.   3.   8.   1.   7.   2.   2.   9.
   5.   4.   7.   2.   3.   4.   5.   5.   4.   0.   5.   5.   4.   5.
   4.   4.   3.   4.   9.   5.   3.   4.   6.   4.   1.   7.   1.   4.
   2.   4.   1.   5.   4.   2.   0.   7.  10.   3.  11.   4.   4.   8.
   1.   3.   4.   2.   2.   7.   2.   8.   9.   6.   4.   2.   4.   8.
   6.   4.   6.   0.   7.   0.   1.   2.   3.   4.   5.   3.   1.   4.
   2.   6.   1.   6.   3.   2.   5.   2.   2.   6.   2.   4.   5.   8.
   8.   2.]]
********************************************************************************
Filename: car_side/test/image_0027.jpg
Ground truth: car_side
Result:       car_side
Histogram:
[[ 4.  12.   2.   3.   4.   7.   8.   1.   8.   8.   5.   5.   3.   3.
   1.   7.   5.   2.   7.   4.   4.   3.   5.   4.   3.   2.   3.   6.
   0.   5.   3.   8.   2.  13.   5.  12.   1.   1.   2.   2.   8.   3.
   2.   3.   1.   9.  11.   6.   4.   8.  12.   6.  13.   4.   7.   1.
   3.   2.   6.   2.   1.   7.   2.   7.  15.   3.   6.   3.   9.   3.
  12.   8.   5.   4.   5.   2.   6.  10.   5.   4.   5.  11.   2.  10.
   8.   4.   6.   4.   7.   1.   4.   3.   3.  12.   2.   6.   6.  13.
   3.   1.]]
********************************************************************************
Filename: car_side/test/image_0028.jpg
Ground truth: car_side
Result:       car_side
Histogram:
[[ 5.   3.  11.   3.  11.   5.  12.  10.   9.   2.   5.   6.   8.   8.
   6.   6.  11.   4.  11.   3.  11.   9.   4.   9.   9.  16.   9.   9.
   3.   9.   6.  12.   8.   5.  11.   8.   6.   9.   2.   7.   4.   2.
   5.   5.   1.   5.  12.  12.   2.  11.   2.   3.  11.   3.   4.  10.
   0.   1.   3.   6.   0.   8.   3.   8.  11.   3.   7.   8.  11.   4.
   7.   3.   6.   4.  12.   3.  11.   4.   2.  10.   3.   5.   8.  10.
   3.  11.   4.   8.   7.   6.   7.   1.   7.  13.   1.   4.   6.   9.
   6.   4.]]
********************************************************************************
```

```
**********************************************************************************
Filename: car_side/test/image_0029.jpg
Ground truth: car_side
Result:       car_side
Histogram:
[[  1.  10.   6.   6.   2.   4.   7.   3.   5.   3.   5.   8.   4.   5.
    5.   8.  10.   4.   4.   3.   5.   4.   1.   5.   7.   7.  13.   4.
    2.   6.   6.   6.   7.   2.   8.  11.   7.   3.   5.   9.   4.   5.
    6.   4.  12.   3.  10.   8.   3.   9.   7.   9.   2.   8.   5.   5.
    1.   5.   4.   1.   2.   1.   3.   3.   5.   6.   7.   6.  10.   5.
    4.   2.   2.   5.   7.   4.   5.   8.   3.   7.   7.   4.   2.   6.
    7.   3.   4.   7.   1.   7.   1.   6.   6.   8.   6.   5.   6.   6.
    2.   5.]]
**********************************************************************************
Filename: car_side/test/image_0030.jpg
Ground truth: car_side
Result:       car_side
Histogram:
[[  2.   4.   2.   5.   2.   5.   8.   4.   5.   4.   1.  18.   8.   2.
    2.   3.  10.   3.   3.   2.   2.   2.   3.   6.   4.   4.   4.   9.
    1.   1.   0.   1.   3.   3.   5.   5.   4.   6.   3.   5.   4.   6.
    4.   5.   4.   4.   4.   3.   3.   5.   4.   1.   3.   2.   5.   2.
    3.   5.  10.   2.   3.   4.   2.   7.   1.   8.   7.   2.   4.   2.
    1.   4.   8.   1.   2.   2.   4.   3.   1.   5.   3.   5.   5.   3.
    5.   3.   2.   5.   6.   5.   2.   2.   6.   6.   3.   2.   3.   0.
    2.   5.]]
car_side detection rate: 1.0
**********************************************************************************
**********************************************************************************
Filename: electric_guitar/test/image_0061.jpg
Ground truth: electric_guitar
Result:       Motorbikes
Histogram:
[[  2.   6.   4.   3.   4.   5.   3.   1.   1.   4.   6.   8.   5.   2.
    1.   2.   5.   5.   4.   2.   2.   6.   5.   4.   4.   2.   3.   5.
    3.   1.   6.   5.   6.   3.   4.   8.   4.   4.   3.   4.   1.  13.
    4.   3.   2.   6.   3.   6.   4.   5.   4.  15.  11.   3.   9.  11.
    2.   1.   2.   3.   5.   9.   4.   2.   4.   1.   4.   1.   3.   2.
    3.   4.  11.   5.   7.   4.   6.   4.   5.   3.   6.   5.   1.   5.
    4.   8.   1.   7.   2.   3.   5.  16.   6.   2.   0.   6.   3.   6.
    1.   2.]]
**********************************************************************************
```

```
********************************************************************************
Filename: electric_guitar/test/image_0062.jpg
Ground truth: electric_guitar
Result:       airplanes
Histogram:
[[  3.   5.   1.   1.   5.   8.   0.   0.   4.   5.   1.  12.   6.   4.
    3.   0.   0.   1.   1.   6.   1.   4.   3.   0.   3.   2.   3.   2.
    1.   0.  10.   0.   1.   1.   1.   0.   0.   6.   6.   1.   2.   8.
    1.   0.   0.   3.   1.   4.   3.   4.   8.  13.   8.   8.   7.   0.
    3.   5.   3.   0.   0.   1.   0.   3.   3.   0.   2.   2.   1.   1.
    0.   0.   1.   5.   5.   1.   4.   4.   3.   1.  16.   6.   1.   2.
    2.   1.   3.   4.   0.   1.   2.   7.   4.   4.   1.   0.   3.   2.
    6.   4.]]
********************************************************************************
Filename: electric_guitar/test/image_0063.jpg
Ground truth: electric_guitar
Result:       Motorbikes
Histogram:
[[  1.   7.   1.   4.   2.   3.   4.   0.   3.   4.   2.   7.   5.   4.
    1.   3.   2.   2.   3.   0.   2.   3.   7.   3.   2.  11.   3.   2.
    1.   1.   1.   1.   1.   4.   2.   6.   0.   2.   3.   1.   2.   4.
    1.   0.   2.   1.   4.   5.   2.   4.   1.   0.   2.   1.   0.   4.
    0.   2.   2.   2.   1.   3.   2.   1.   7.   1.   2.   3.   3.   1.
    1.   1.   1.   0.   3.   0.   4.   1.   3.   5.   8.   0.   0.   1.
    2.   4.   1.   3.   2.   1.   3.   3.   1.   1.   0.   2.   3.   5.
    7.   1.]]
********************************************************************************
Filename: electric_guitar/test/image_0064.jpg
Ground truth: electric_guitar
Result:       electric_guitar
Histogram:
[[  0.   3.   0.   1.   5.   1.   7.   3.   2.   2.   2.  10.   8.   2.
    0.   1.   1.   0.   2.   5.   1.   3.   1.   2.   3.   2.   3.   0.
    1.   2.   3.   0.   0.   0.   2.   2.   3.   2.   2.   1.   2.   3.
    1.   0.   1.   0.   6.   1.   3.   0.   2.   5.   0.   3.   2.   0.
    3.   4.   2.   0.   0.   2.   8.   5.   1.   0.   1.   1.   1.   0.
    1.   0.   1.   1.   3.   0.   1.   2.   1.   3.  11.   2.   2.   3.
    2.   4.   0.   0.   3.   1.   0.   8.   5.   2.   2.   0.   2.   2.
    2.   0.]]
********************************************************************************
```

```
********************************************************************************
Filename: electric_guitar/test/image_0065.jpg
Ground truth: electric_guitar
Result:       electric_guitar
Histogram:
[[  3.   1.   1.   2.   0.   2.   3.   2.   2.   3.   3.  14.   1.   9.
    4.   2.   3.   6.   2.   0.   2.   6.   2.   2.   1.   1.   1.   0.
    1.   2.   0.   0.   0.   1.   1.   8.   1.   6.   0.   2.   3.   1.
    2.   1.   2.   3.   1.   0.   0.   0.   1.   3.   4.   1.   3.  11.
    0.   1.   3.   0.   1.   4.   6.   2.   6.   0.   2.   0.   1.   2.
    2.   2.   3.   4.   4.   1.   0.   1.   0.   1.   5.   4.   1.   0.
    0.   2.   4.   0.   0.   2.   9.   1.   1.   0.   2.   6.   4.   3.
    3.   1.]]
********************************************************************************
Filename: electric_guitar/test/image_0066.jpg
Ground truth: electric_guitar
Result:       Motorbikes
Histogram:
[[ 1.  3.  2.  0.  3.  3.  5.  2.  0.  0.  4.  9.  0.  6.  1.  2.  3.  0.
   3.  1.  2.  3.  4.  2.  3.  2.  1.  3.  3.  2.  4.  0.  4.  0.  2.  1.
   3.  1.  2.  0.  1.  5.  2.  0.  1.  1.  4.  1.  1.  3.  5.  1.  5.  4.
   0.  1.  2.  0.  1.  0.  2.  1.  0.  3.  2.  5.  2.  1.  1.  1.  0.  4.
   1.  5.  1.  1.  2.  0.  1.  1.  2.  1.  1.  3.  1.  3.  5.  3.  2.  2.
   5.  1.  4.  1.  2.  4.  3.  8.  1.  3.]]
********************************************************************************
Filename: electric_guitar/test/image_0067.jpg
Ground truth: electric_guitar
Result:       electric_guitar
Histogram:
[[  1.   4.   1.   1.   1.   0.   4.   1.   1.   3.   0.  14.   6.   2.
    3.   1.   0.   0.   1.   2.   5.   5.   2.   1.   2.   1.   1.   2.
    0.   0.   3.   1.   2.   3.   0.   3.   2.   1.   4.   2.   2.   3.
    0.   0.   0.   5.   3.   1.   0.   2.   3.   6.   1.   4.   1.   3.
    1.   1.   3.   0.   2.   1.   7.   1.   5.   0.   0.   0.   1.   1.
    0.   1.   3.   3.   1.   1.   2.   1.   1.   1.  13.   0.   0.   3.
    0.   1.   0.   1.   3.   2.   3.   3.   3.   2.   0.   3.   1.   2.
    3.   4.]]
********************************************************************************
```

```
********************************************************************************
Filename: electric_guitar/test/image_0068.jpg
Ground truth: electric_guitar
Result:       airplanes
Histogram:
[[  7.   5.   4.   5.   1.   1.   4.   1.   1.   6.   6.  15.   9.   4.
    4.   7.   2.  10.   5.   3.   4.   8.   9.   8.   6.   8.   0.   4.
    3.  11.   9.   2.   4.   5.   2.   4.   6.   3.   6.   4.   1.  20.
    7.   9.   2.   3.   4.   1.   1.   1.   2.   6.  18.   5.   2.   8.
    5.   7.   8.   1.  18.   2.  10.   8.   4.   8.   2.   3.   5.   3.
    0.   1.   8.   2.   5.   3.   3.   5.   3.   5.  13.   4.   0.   7.
    0.   6.  11.   5.   5.   6.   3.   6.  14.   5.   8.   4.   3.   3.
    3.   8.]]
********************************************************************************
Filename: electric_guitar/test/image_0069.jpg
Ground truth: electric_guitar
Result:       airplanes
Histogram:
[[  3.   4.   4.   4.   6.   4.   4.   5.   3.   7.  10.  18.   4.   7.
   10.   9.   6.   3.   6.   3.   4.   8.  11.   4.   4.   8.   3.   4.
    2.   2.  11.   1.   3.   7.   1.   7.   2.   5.  13.   2.   3.   9.
    3.   8.   1.   5.   3.   6.   2.   3.   1.   7.   9.   6.   4.   2.
   11.   7.   5.   0.   9.   5.   7.   8.   5.  11.   7.   2.   5.   3.
    2.   2.   5.   2.   7.   5.   4.   4.   1.   3.  15.   2.   2.   3.
    2.   4.  13.   6.   2.   4.   5.  14.  12.   4.   3.   5.   6.   2.
    7.   8.]]
********************************************************************************
Filename: electric_guitar/test/image_0070.jpg
Ground truth: electric_guitar
Result:       Motorbikes
Histogram:
[[  1.   0.   3.   2.   2.   1.   3.   1.   1.   2.   1.   1.   0.   1.
    0.   1.   6.   1.   1.   1.   4.  23.   3.   5.   2.   7.   2.   3.
    4.   0.   0.   3.   3.   2.   0.   3.   0.   0.   0.   1.   2.   4.
    1.   1.   2.   2.   3.   0.   1.   2.   0.   2.  22.   2.   1.   2.
    2.   2.   1.   1.   0.   1.   0.   0.   1.   0.   6.   0.   2.   0.
    1.   2.   2.  16.   1.   1.   3.   2.   0.   2.   5.   1.   0.   1.
    1.   3.   7.   2.   0.   2.   2.   4.   1.   2.   1.   0.   1.   3.
    3.   4.]]
electric_guitar detection rate: 0.3
********************************************************************************
```

```
********************************************************************************
Filename: faces/test/image_0261.jpg
Ground truth: faces
Result:       faces
Histogram:
[[ 1.   2.   2.   1.   3.   2.   3.   2.   2.   1.   0.   8.   7.   1.
   2.   1.   4.   2.   2.   2.   8.   0.   1.   3.   2.   5.   3.   3.
   4.   3.   0.   0.   3.   2.   4.   3.   1.   7.   4.   1.   1.   3.
   6.   2.   2.   2.   2.   5.   4.   2.   2.   6.   0.   2.   3.   3.
   1.   1.   4.   4.   1.   3.   2.   4.   3.   4.   2.   6.   2.   2.
   1.   3.   0.   2.   3.   3.   1.   0.   1.   6.   7.   5.   0.   3.
   1.   0.   2.   2.   4.   3.   5.   7.  10.   3.   3.   0.   2.   1.
   0.   1.]]
********************************************************************************
Filename: faces/test/image_0274.jpg
Ground truth: faces
Result:       car_side
Histogram:
[[ 3.   5.   2.  12.   6.   4.   4.   2.  10.   5.   2.   3.   4.   4.
   5.   3.   6.   7.   4.   8.   3.   2.   4.   5.   3.   3.   5.   5.
   5.   4.   2.   8.   3.   9.  11.   7.   4.   5.   5.  12.   6.   2.
  10.   1.   3.   5.   2.   5.   4.   7.   9.   4.   5.   7.   2.  17.
   3.   4.   3.   1.   6.  13.   4.   2.  10.   5.   6.   5.   6.   4.
   5.   7.   8.   2.   5.   3.   5.   8.   2.   9.   6.   3.   5.   9.
   0.   4.   2.   5.   8.   5.   8.   7.   2.   6.   1.  10.   5.   9.
   5.   2.]]
********************************************************************************
Filename: faces/test/image_0287.jpg
Ground truth: faces
Result:       faces
Histogram:
[[ 2.   1.   1.   5.   4.   2.   4.   2.   1.   5.   4.  15.   2.   2.
   4.   1.   3.   3.   0.   2.   1.   2.   0.   9.   0.   9.   3.   3.
   4.   3.   5.   1.   1.   2.   2.   1.   1.   3.   4.   0.   1.   4.
   3.   3.   9.   5.   1.   3.   1.   2.   6.   2.   2.   3.   2.   2.
   0.   2.   5.   1.   2.   2.   1.   3.   7.   0.   1.   2.   3.   1.
   1.   0.   2.  14.   5.   1.   1.   4.   1.   3.   8.   0.   3.   2.
   3.   2.   1.   4.   4.   4.   3.   7.   7.   2.   2.   1.   1.   4.
   1.   2.]]
********************************************************************************
```

```
********************************************************************************
Filename: faces/test/image_0300.jpg
Ground truth: faces
Result:       car_side
Histogram:
[[  6.   4.   8.  10.   7.   5.   7.   5.  14.   5.   6.  11.   6.   7.
    7.   6.  12.   4.   3.   2.  11.   5.   4.   4.   9.  15.  17.   6.
    2.   5.   1.   9.   2.   6.  10.  15.   4.   8.   5.  11.   6.   4.
    6.   6.   2.   6.  15.   4.   1.  10.  10.   0.   5.   1.   7.  16.
    4.   3.   5.   1.   1.   5.   2.   7.   9.  12.  10.   8.  13.   8.
   12.   3.   9.   9.   3.   3.   9.  12.   1.   5.  16.   7.   8.   9.
    6.   6.   9.   5.   3.  12.   9.   5.   5.  12.   1.   4.  10.   3.
    0.   6.]]
********************************************************************************
Filename: faces/test/image_0313.jpg
Ground truth: faces
Result:       faces
Histogram:
[[  5.   7.   6.  10.   7.   2.   7.   8.   5.   4.   3.   3.   7.   1.
    4.   4.   5.   2.   5.  11.   5.   2.   3.   3.   4.   9.   4.   5.
    5.   8.   7.   2.   1.   2.   5.   7.   2.   8.   4.   3.   2.   8.
    3.   0.   6.   5.   6.   3.   5.   6.   0.  21.   7.   3.   2.  12.
    2.   2.   4.   5.   4.   6.   4.   3.   9.  12.   2.   4.   5.   7.
    2.   2.   2.   9.   1.   5.   5.   6.   1.  11.  18.   8.   2.   1.
    3.   2.   5.   1.   3.   2.   3.  19.   3.   5.   7.   6.   4.   5.
    3.   1.]]
********************************************************************************
Filename: faces/test/image_0326.jpg
Ground truth: faces
Result:       faces
Histogram:
[[  4.   4.   3.   2.   3.   0.   3.   5.   2.   3.   2.  26.   1.   0.
    1.   4.   3.   4.   4.   0.   2.   1.   8.   0.   3.   4.   3.   1.
    5.   5.   6.   1.   2.   0.   4.   3.   6.   0.   1.   1.   1.   3.
    1.   1.  14.   1.   1.   4.   4.   1.   2.  10.   2.   7.   0.   7.
    6.   3.  13.   1.   6.   2.   0.   0.   3.   1.   2.   5.   0.   2.
    5.   1.   3.   8.   1.   1.   2.   6.   2.   4.  16.   1.   4.   3.
    0.   5.   1.   3.   5.   5.   5.   6.   3.   0.   5.  12.   1.   1.
    3.   6.]]
********************************************************************************
```

```
********************************************************************************
Filename: faces/test/image_0339.jpg
Ground truth: faces
Result:       Motorbikes
Histogram:
[[  6.   2.   1.   7.   3.   1.   3.   6.   3.   9.   4.   8.   7.   4.
    3.   6.   1.   5.   4.   5.   2.   5.   6.   7.   5.   5.   5.   3.
   10.  10.  11.   3.   4.   4.   1.   7.   4.   2.   7.   1.   4.   5.
    6.   2.   4.   9.   4.   4.   5.   4.   4.  16.   2.   7.   2.  11.
    3.   4.  14.   2.   5.   8.   5.   5.   3.   6.   5.   4.   3.   5.
    4.   8.   3.   6.   4.   5.   4.   2.   3.   2.  12.   4.   3.   6.
    2.   8.   1.   6.   7.  12.  10.  14.   9.   6.   8.   2.   1.   9.
    9.  11.]]
********************************************************************************
Filename: faces/test/image_0352.jpg
Ground truth: faces
Result:       faces
Histogram:
[[  1.   1.   2.   3.   2.   0.   0.   1.   1.   4.   5.  21.   2.   8.
    3.   5.   3.   3.   1.   3.   1.  14.   9.   3.   1.   1.   2.   1.
    2.   1.   6.   1.   2.   1.   3.   0.   2.   1.   5.   4.   0.   8.
    7.   2.   3.   3.   3.   3.   2.   2.   2.   1.   7.  10.   3.   5.
    1.   4.  16.   0.   0.   2.   2.   2.   1.   1.   0.   1.   3.   2.
    3.   2.   0.   8.   1.   1.   1.   3.   3.   4.  19.   2.   1.   7.
    3.   0.   7.   0.   1.   0.   2.   4.   7.   1.   1.   5.   4.   1.
    3.   2.]]
********************************************************************************
Filename: faces/test/image_0365.jpg
Ground truth: faces
Result:       faces
Histogram:
[[  2.   5.   3.   4.   9.   4.   4.   6.   3.   3.   6.  24.   5.   2.
    3.   5.   9.   6.   5.   5.   2.   7.   3.   5.   4.   4.   7.   3.
   15.   6.  14.   1.   5.   7.  11.   1.   7.   4.   7.   5.   4.  11.
    7.   1.   6.   9.   4.   7.   3.   3.   5.   5.   5.   9.   6.   2.
    3.   6.  11.   2.   6.   2.   0.   3.   4.   8.   6.   7.   2.   1.
   10.   2.   4.  11.   4.   4.   4.   5.   3.   5.  19.   7.   3.  10.
    6.   4.   3.   5.   2.   7.   3.  18.   7.   2.   6.   8.   5.   6.
    6.   4.]]
********************************************************************************
```

```
********************************************************************************
Filename: faces/test/image_0378.jpg
Ground truth: faces
Result:       faces
Histogram:
[[  2.   5.   2.   2.   3.   5.   4.   1.   4.   2.   3.  11.   3.   2.
    0.   2.   4.   3.   0.   3.   0.   1.   4.   0.   0.   1.   0.   1.
    3.   5.   9.   0.   6.   2.   5.   0.   5.   2.   4.   3.   1.   4.
    6.   0.   5.   2.   3.   5.   3.   7.   4.  15.   3.   2.   1.   7.
    0.   7.   7.   3.   3.   3.   0.   2.   2.   1.   1.   1.   2.   1.
    3.   2.   2.  10.   4.   3.   0.   0.   1.   3.  10.   1.   2.   4.
    3.   1.   1.   4.   1.   2.   1.   9.   7.   2.   5.   0.   5.   0.
    3.   2.]]
faces detection rate: 0.7
********************************************************************************
********************************************************************************
Filename: Motorbikes/test/image_0101.jpg
Ground truth: Motorbikes
Result:       airplanes
Histogram:
[[  0.   4.   4.   2.   2.   2.   2.   2.   2.   1.   4.  32.   4.  12.
    3.   2.  12.   1.   6.   3.   3.   5.   1.   2.   6.   5.   1.   8.
    3.   5.   4.   1.   4.   5.   5.   4.   3.   7.   1.   1.   5.   1.
    0.   2.   0.   3.   5.   2.   4.   2.   2.   9.   4.   1.   1.  10.
    4.   4.   1.   2.   5.   7.   4.   3.   4.   4.   6.   7.   5.   9.
    4.   4.   1.   0.   6.   6.   3.   3.   1.  10.   3.   3.   5.   2.
    1.   3.   1.   2.   4.   4.   5.   9.   1.   0.  11.   2.   1.   3.
    1.   1.]]
********************************************************************************
Filename: Motorbikes/test/image_0102.jpg
Ground truth: Motorbikes
Result:       car_side
Histogram:
[[  1.   1.   3.  13.   2.   0.   4.   1.   2.   8.   5.  16.   2.   7.
    6.   9.   1.   2.  11.   0.   1.   8.   1.   4.   5.   4.   5.  11.
    1.   6.   0.   2.   0.   6.   1.   6.   2.   1.   1.   8.   3.   3.
    8.   3.   3.   0.   4.   0.   0.   1.   4.   0.  20.   0.   3.   9.
    9.   1.   5.   2.   4.   5.   5.   7.   3.   9.   1.   2.   3.   5.
    0.   7.   6.   0.   6.   1.   1.   6.   0.   1.   5.   3.   2.   2.
    1.   4.   4.   3.   2.   7.  11.   0.   2.   4.   5.   9.   2.   4.
    1.   6.]]
********************************************************************************
```

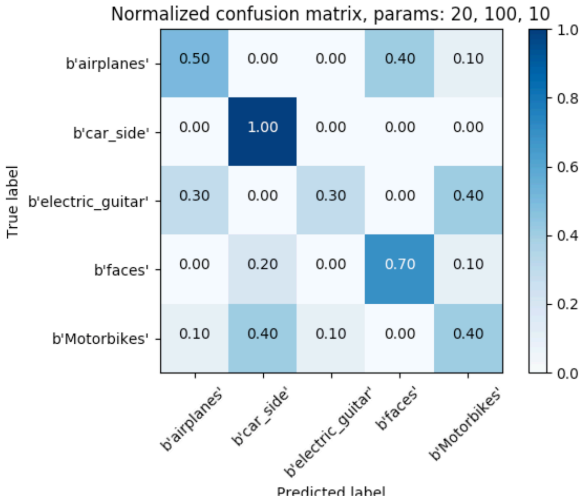
```
********************************************************************************
Filename: Motorbikes/test/image_0103.jpg
Ground truth: Motorbikes
Result:       Motorbikes
Histogram:
[[  3.   5.   5.   8.   0.   5.   5.   6.   9.   2.  12.  17.   8.   9.
    4.   8.   7.   6.  10.   6.  11.   6.   0.  10.   6.  11.   5.  15.
    6.   8.   1.   0.   2.   4.   8.   1.   3.   3.   5.   3.   3.   2.
    5.   2.   3.   6.   8.   5.   1.   8.   4.   0.   6.   2.   5.   3.
   15.   2.  13.   5.  12.   7.  12.   4.   3.   7.   6.   3.  11.   5.
    7.   7.   1.   1.  11.   7.   6.   8.   6.   5.   8.   3.   7.   2.
    1.   7.   6.   7.   0.  13.   9.   2.   2.   5.  10.   7.   3.   1.
    5.   6.]]
********************************************************************************
Filename: Motorbikes/test/image_0104.jpg
Ground truth: Motorbikes
Result:       Motorbikes
Histogram:
[[  3.   2.   1.   3.   1.   2.   2.   2.   1.   2.   4.  14.   5.   6.
    3.   8.   1.   9.   4.   4.   1.   7.   0.   5.   1.   2.   7.   8.
    5.   6.   0.   3.   3.   2.   0.   2.   3.   5.   0.   2.   4.   0.
    6.   2.   3.   2.   5.   1.   0.   1.   1.   5.   8.   1.   1.   8.
    1.   1.   6.   5.   9.   6.   2.   2.   5.  11.   4.   4.   4.   2.
    4.   6.   2.   4.   7.   1.   5.   2.   3.   2.   8.   0.   4.   1.
    2.   2.   1.   1.   1.   5.   6.   1.   2.   3.   8.   4.   4.   1.
    0.   4.]]
********************************************************************************
Filename: Motorbikes/test/image_0105.jpg
Ground truth: Motorbikes
Result:       car_side
Histogram:
[[  3.   5.   2.   3.   1.   1.   0.   2.   1.   0.   2.   8.   3.   7.
    4.   5.   4.   6.   3.   3.   4.   7.   1.   2.   3.   0.   3.   8.
    4.   2.   1.   2.   1.   1.   2.   2.   3.   6.   2.   4.   1.   2.
    4.   1.   6.   0.   6.   1.   1.   2.   1.   0.   7.   3.   1.   4.
    1.   1.  10.   0.  22.   4.   1.   1.   1.   3.   2.   4.   3.   4.
    6.   4.   3.   0.   7.   1.   0.   1.   2.   1.   3.   8.   2.   2.
    1.   4.   3.   1.   4.   2.  13.   1.   6.   4.   2.   8.   3.   2.
    1.   2.]]
********************************************************************************
```

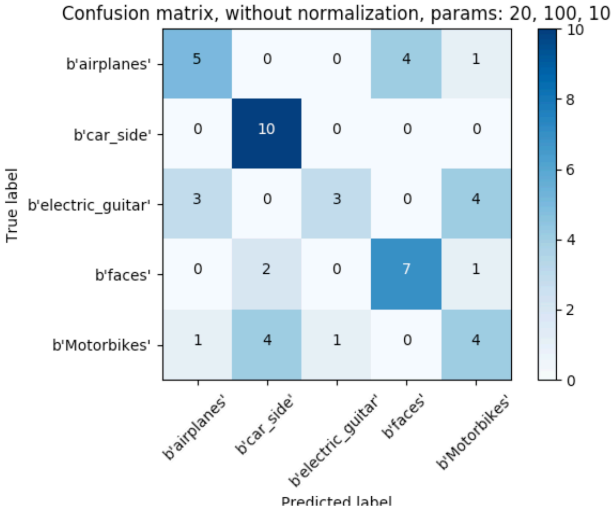
```
********************************************************************************
Filename: Motorbikes/test/image_0106.jpg
Ground truth: Motorbikes
Result:       car_side
Histogram:
[[  2.   7.   2.   6.   2.   0.   9.   1.   0.   5.   5.   7.   8.   5.
    5.   3.   7.   2.  10.   1.   5.   2.   4.   5.   0.   5.   7.   1.
    6.   4.   0.   3.   4.   4.   2.   8.   1.   4.   1.   2.   2.   2.
    2.   6.  12.   3.   8.   0.   0.   4.   2.   2.   3.   3.   4.   8.
    7.   1.   5.   2.   6.   7.   2.   3.   7.   2.   6.   6.   7.   4.
    9.   4.   5.   6.   9.   0.   1.   6.   3.   0.   4.   5.   2.   0.
    2.   0.   0.   5.   5.   6.   5.   6.   3.   2.   3.   3.   2.   1.
    3.   5.]]
********************************************************************************
Filename: Motorbikes/test/image_0107.jpg
Ground truth: Motorbikes
Result:       car_side
Histogram:
[[  2.   5.   3.   3.   5.   1.   5.   2.   3.   5.   3.  13.   4.   2.
    7.   5.   4.   0.   6.   1.   9.   7.   5.  11.   6.   6.   6.  10.
    5.   6.   1.   5.   5.   7.   6.   6.   6.   6.   2.   8.   3.   2.
    9.   2.   5.   3.   4.   3.   1.   8.   3.   7.  10.   4.   2.  10.
   11.   2.   8.   4.   1.   4.  10.   2.   7.   9.   5.   7.   4.   9.
    7.   1.   5.   2.   3.   5.   6.   4.   0.   4.   2.   3.   4.   2.
    3.   3.   2.   5.   4.  10.  11.   9.   6.   2.   7.   6.   2.   5.
    5.   8.]]
********************************************************************************
Filename: Motorbikes/test/image_0108.jpg
Ground truth: Motorbikes
Result:       electric_guitar
Histogram:
[[ 3.  1.  2.  3.  1.  2.  3.  1.  2.  3.  1.  9.  3.  7.  2.  0.  0.  3.
   1.  0.  4.  2.  1.  2.  0.  1.  2.  1.  2.  1.  2.  6.  0.  0.  2.  4.
   2.  1.  3.  3.  4.  2.  2.  1.  5.  2.  2.  1.  2.  1.  6.  3.  1.  0.
   2.  8.  1.  2.  4.  0.  1.  1.  0.  3.  5.  0.  1.  2.  4.  5.  4.  0.
   9.  3.  8.  1.  5.  0.  1.  0.  7.  1.  0.  2.  2.  3.  1.  6.  1.  3.
   1.  0.  8.  2.  0.  1.  0.  3.  1.  5.]]
********************************************************************************
```

```
Number of components: 20
Number of clusters: 100
Number of neighbors: 10
Correct number: 29
Test number: 50
Accuracy:  0.58

Confusion matrix, without normalization
[[ 5  0  0  4  1]
 [ 0 10  0  0  0]
 [ 3  0  3  0  4]
 [ 0  2  0  7  1]
 [ 1  4  1  0  4]]


Normalized confusion matrix
[[ 0.5  0.   0.   0.4  0.1]
 [ 0.   1.   0.   0.   0. ]
 [ 0.3  0.   0.3  0.   0.4]
 [ 0.   0.2  0.   0.7  0.1]
 [ 0.1  0.4  0.1  0.   0.4]]
```





```
Number of components: 20
Number of clusters: 100
Number of neighbors: 12
Correct number: 28
Test number: 50
Accuracy:  0.56

Number of components: 20
Number of clusters: 100
Number of neighbors: 14
Correct number: 28
Test number: 50
Accuracy:  0.56

Number of components: 20
Number of clusters: 120
Number of neighbors: 10
Correct number: 32
Test number: 50
Accuracy:  0.64

Number of components: 20
Number of clusters: 140
Number of neighbors: 10
Correct number: 33
Test number: 50
Accuracy:  0.66

Number of components: 22
Number of clusters: 100
Number of neighbors: 10
Correct number: 33
Test number: 50
Accuracy:  0.66

Number of components: 24
Number of clusters: 100
Number of neighbors: 10
Correct number: 34
Test number: 50
Accuracy:  0.68
```

More experiments:

3. A summary and discussion of the results, including effects of parameter choices.

Page 2 - 17 shows my classification results with Number of components to be 20, Number of clusters to be 100, and Number of neighbors to be 10 (suggested in the assignment.) It shows some images has better result than the others. For example, car_side has 100% accuracy, but electric guitar only has 30% accuracy. I also did some experiments by using different number of components, clusters and neighbors.

It shows with higher number of clusters, the accuracy of classification is higher than that of for lower number of clusters. The accuracy can achieve 66% by using 140 clusters.

It shows with higher number of neighbors, the accuracy does not change.

It shows with higher number of components, the accuracy of classification is higher than that of for lower number of components. The accuracy can achieve 68% by using 24 components.

```
import itertools
import cv2
import numpy as np
import os
from sklearn.metrics import confusion_matrix
import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt

# plot the confusion matrix
# http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    print(cm)

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

# Get Vector for train image
def get_vector(NUM_CLUSTERS, labels):
    v = np.zeros(NUM_CLUSTERS, dtype="float32")
    for ii in labels:
        v[ii]=v[ii]+1
    return v

# Get Vector for test image
def get_vector_test(NUM_CLUSTERS, test_pca, centers):
    v = np.zeros(NUM_CLUSTERS, dtype="float32")
    for feat in test_pca:
        dist = feat - centers[0]
        min_length = sum(dist*dist)**0.5
        min_index = 0
        i = 0
        for center in centers:
            dist = feat-center
```

```
        if train_f is None: train_f = j
        else: train_f = np.vstack((train_f, j))

test_f = None
for i in test_descriptor:
    for j in i:
        if test_f is None: test_f = j
        else: test_f = np.vstack((test_f, j))

initial_train_f_size = train_f.shape
initial_test_f_size = test_f.shape

# Get PCA, mean and eigenvector
pca_avg, pca_eigen = cv2.PCACompute(train_f, None)
# Use 20 components
pca_eigen_selected = pca_eigen[:NUM_COMPONENTS]

#Print Info for assignment
print( "pca_avg shape:", pca_avg.shape)
print ("pca_eigen shape", pca_eigen.shape)
print ("pca_avg:")
print (pca_avg)
print ("pca_eigen")
print (pca_eigen)


# project Train and Test features
train_f_pca = np.zeros((initial_train_f_size[0], NUM_COMPONENTS))
for i in range(len(train_f)):
    train_f_pca[i] = np.dot(pca_eigen_selected, train_f[i] - pca_avg[0])
test_f_pca = np.zeros((initial_test_f_size[0], NUM_COMPONENTS))
for i in range(len(test_f)):
    test_f_pca[i] = np.dot(pca_eigen_selected, test_f[i] - pca_avg[0])


# convert from numpy array to list
train_des_pca = [[] for i in range(5)]
train_labels_lst=[]
p1 = 0 ; p2 = 0
for i in range(len(train_descriptor)):
    for j in range(len(train_descriptor[i])):
        p2 = len(train_descriptor[i][j]) + p2
        train_des_pca[i].append(train_f_pca[p1:p2])
        train_labels_lst.append(i)
        p1 = p2
train_labels = np.vstack(train_labels_lst).astype(np.float32)

test_des_pca = [[] for i in range(5)]
test_labels_lst = []
p1 = 0 ; p2 = 0
for i in range(len(test_descriptor)):
    for j in range(len(test_descriptor[i])):
        p2 = len(test_descriptor[i][j]) + p2
        test_des_pca[i].append(test_f_pca[p1:p2])
        test_labels_lst.append(i)
        p1 = p2
test_labels = np.vstack(test_labels_lst).astype(np.float32)

# kmeans algorithm
```

```
# kmeans algorithm
criter = (cv2.TERM_CRITERIA_EPS, 1000000000, 0.1)
ret, best_label, c = cv2.kmeans(train_f_pca.astype(np.float32), NUM_CLUSTERS, None, criter, 20, cv2.
    KMEANS_RANDOM_CENTERS)

print ('retval:')
print (ret)
print ('best_labels:')
print (best_label)
print ('centers:')
print (c)

# get the histogram for each training image
train_his = np.array([],dtype="float32")
train_his.shape=(0, NUM_CLUSTERS)
p1 = 0 ; p2 = 0
for i in range(len(train_des_pca)):
    for j in range(len(train_des_pca[i])):
        p2 = len(train_des_pca[i][j]) + p2
        his = get_vector(NUM_CLUSTERS, best_label[p1:p2])
        train_his = np.append(train_his, [his], axis=0)
        p1 = p2
print ("train_histo shape:", train_his.shape)
print ('Histogram of first training image:')
print (train_his[0])

# initite the knn class
knn = cv2.ml.KNearest_create()
knn.train(train_his, cv2.ml.ROW_SAMPLE, train_labels)

# Test Iamge: histogram,label, accuracy
result_labels_lst = []
num_test = 0
num_correct = 0
for i in range(len(test_des_pca)):
    print ('*********************************************************************************')
    num_correct_cate = 0
    for j in range(len(test_des_pca[i])):
        his = get_vector_test(NUM_CLUSTERS, test_des_pca[i][j], c)
        ret2, result, neighbor, dist = knn.findNearest(his, NUM_NEIGHBORS)
        num_test += 1
        result_labels_lst.append(result)
        print ('*********************************************************************************')
        print ("Filename:", test_name[i][j].split("HW5_Data/")[1])
        print ("Ground truth:", label_class[i])
        print ("Result:      ", label_class[int(result[0][0])])
        print ('Histogram:')
        print (his)

        if int(result[0][0]) == i:
            num_correct += 1
            num_correct_cate += 1

    print (label_class[i], "detection rate:", float(num_correct_cate)/len(test_des_pca[i]))


# Final Result
print ('*********************************************************************************')
print ('\n')
```

```
# Final Result
print ('*********************************************************************************')
print ('\n')
print ("Number of components:", NUM_COMPONENTS)
print ("Number of clusters:", NUM_CLUSTERS)
print ("Number of neighbors:", NUM_NEIGHBORS)
print ("Correct number:", num_correct)
print ("Test number:",num_test)
print ("Accuracy: ", float(num_correct)/num_test)
print ('\n')

# Confusion matrix
class_names_lst = [label_class[0], label_class[1], label_class[2], label_class[3], label_class[4]]
result_labels = np.vstack(result_labels_lst).astype(np.float32)
class_names = np.squeeze(np.vstack(class_names_lst).astype(np.string_))
cnf_matrix = confusion_matrix(np.squeeze(test_labels.astype(np.int)), np.squeeze(result_labels.astype(np.int)))

# print the confusion matrix
np.set_printoptions(precision=2)
params = ", params: "+str(NUM_COMPONENTS) + ", " + str(NUM_CLUSTERS) + ", " + str(NUM_NEIGHBORS)
print ('\n')

# Plot non-normalized confusion matrix
plt.figure()
plot_confusion_matrix(cnf_matrix, classes=class_names,
                      title='Confusion matrix, without normalization'+params)
print ('\n')

# Plot normalized confusion matrix
plt.figure()
plot_confusion_matrix(cnf_matrix, classes=class_names, normalize=True,
                      title='Normalized confusion matrix'+params)
print ('\n')
plt.show()
```