CS498 Homework 3

Xinchen Pan, Fangxiaozhi Yu, Jiefei Shi

February 18, 2017

4.10

(a)

For this problem, we used the CIFAR-10 dataset from https://www.cs.toronto.edu/~kriz/cifar.html. Since we chose to use python for this assignment, we downloaded the python version of the dataset. It consists of 60000 32×32 color images in 10 classes, wth 6000 images per class. There are 50000 training images and 10000 test images. However, as we are not going to do predictions, we combined all 5 training dataset and 1 testing dataset to form a large 60000×3073 dimension dataset.

We used `unpickle` function to load the dataset, then we separated the dataset by **X** which indicates the data of the images and **y** which indicates the label. **X** has 3072 attributes which are the values of the pixels of the images. **y** is the label of the image and its values are the integers between $0 - 9$. Because the values of pixels are between 0 and 255, **I scaled X by dividing 255**. We would show the both the scaled error and unscaled error, however, all other results would be the scaled version. **(Note: In order to get the unscaled data, simply delete the $X = X/255$ in the code file.)**

In order to find out the principal components for each cateogry, we need to find out the mean for each category first. The following data is a 10×3072 dimension which contains the mean for each column for each category.

```
######mean for each column for each cateogry, 10 by 3072######
array([[ 0.57045033,  0.56641111,  0.5690902 , ...,  0.54348039,
         0.54253268,  0.54447647],
       [ 0.5360719 ,  0.53508954,  0.54002288, ...,  0.48685817,
         0.49044706,  0.49444771],
       [ 0.4888902 ,  0.48889412,  0.49301046, ...,  0.42073137,
         0.41933856,  0.41989412],
       ...,
       [ 0.53094248,  0.52604248,  0.52878889, ...,  0.41724837,
         0.41886013,  0.42390915],
       [ 0.57366013,  0.57014052,  0.57390588, ...,  0.45278627,
         0.45694183,  0.46260784],
       [ 0.65378889,  0.65011765,  0.65382353, ...,  0.49645033,
         0.50066471,  0.5059098 ]])
```

Next, we separated data by their label. Since we have 10 categories, each category of data is $(6000, 3072)$ dimension. We then formed a new dataset by translating our data to zero mean using $x_i - \text{mean}(\{x\})$.

We next used functions from `scikit-learn` to get our principal components. We can easily achieve them by using

```
pca = PCA()
pca.fit(X) #X is our data
pca.components_
```

```
                         #####First 20 principal components#####
[array([[-0.02068911, -0.02053497, -0.02056297, ..., -0.02322361,
         -0.02313471, -0.02321534],
        [ 0.02788187,  0.02811101,  0.02860207, ..., -0.02769933,
         -0.0274672 , -0.02723467]]),
  array([[-0.02945004, -0.02928557, -0.02930782, ..., -0.0205087 ,
         -0.02065396, -0.0208163 ],
        [ 0.01115761,  0.01095572,  0.01122535, ..., -0.03056617,
         -0.02945514, -0.02846108]]),
  array([[-0.02249444, -0.02241051, -0.02229355, ..., -0.02004589,
         -0.02037377, -0.02069153],
        [ 0.01669615,  0.01711018,  0.01786196, ..., -0.02133284,
         -0.02065154, -0.01998308]]),
  array([[-0.02529398, -0.02528944, -0.02558019, ..., -0.022302  ,
         -0.02251589, -0.02282689],
        [-0.02432488, -0.02476216, -0.02481837, ...,  0.0117369 ,
          0.01032032,  0.00907964]]),
  array([[ 0.02664151,  0.0266163 ,  0.02663838, ...,  0.01339896,
          0.01354067,  0.01388686],
        [ 0.0158059 ,  0.01643477,  0.01702466, ..., -0.02709309,
         -0.02684511, -0.0263826 ]]),
  array([[ 0.02555281,  0.02543965,  0.02537582, ...,  0.02144806,
          0.02158175,  0.02203451],
        [ 0.0215726 ,  0.02187737,  0.02210392, ..., -0.00176638,
         -0.00076293,  0.00031488]]),
  array([[ 0.02712535,  0.02715843,  0.02728446, ...,  0.02493408,
          0.02488517,  0.02506184],
        [-0.02092755, -0.02117561, -0.0216773 , ...,  0.01500083,
          0.01439826,  0.01367213]]),
  array([[ 0.03344285,  0.03343548,  0.03350003, ...,  0.00884698,
          0.00934869,  0.00972852],
        [-0.01187202, -0.01301205, -0.01385586, ...,  0.02839487,
          0.02822463,  0.0278939 ]]),
  array([[ 0.02768825,  0.02777998,  0.02791284, ...,  0.01265471,
          0.01294995,  0.01308668],
        [ 0.00754132,  0.00805689,  0.00840478, ..., -0.0213466 ,
         -0.0204671 , -0.01988264]]),
  array([[ 0.03040126,  0.03057937,  0.03069863, ...,  0.01858413,
          0.01872055,  0.01906443],
        [-0.01430363, -0.01486679, -0.01516347, ...,  0.0342604 ,
          0.03365492,  0.03312609]])]
```

In order to reconstruct the data , we used function `pca.transform` to apply dimensionality reduction to X then follow the following

$$\hat{x}_i = Up_i + \text{mean}(\{x\})$$

To calucalte the error, we used

$$||x_i - \hat{x}_i||^2$$

The following is the error resulting from the images of each category using the first 20 princial components against category

Table 1: Error of Each Category scaled/unscaled

category	scaled error	unscaled error
airplain	241799.4	15723005140
automobile	364537.6	23704055030
bird	225854.5	14686186486
cat	287564.4	18698873336
deer	201189.5	13082346951
dog	298141.9	19386680261
frog	242698.4	15781464395
horse	317517.1	20646547740
ship	225202.8	14643809661
truck	371035.2	24126563747

The error plot resulting from representing the images of each category using the first 20 principal components against the category is below.

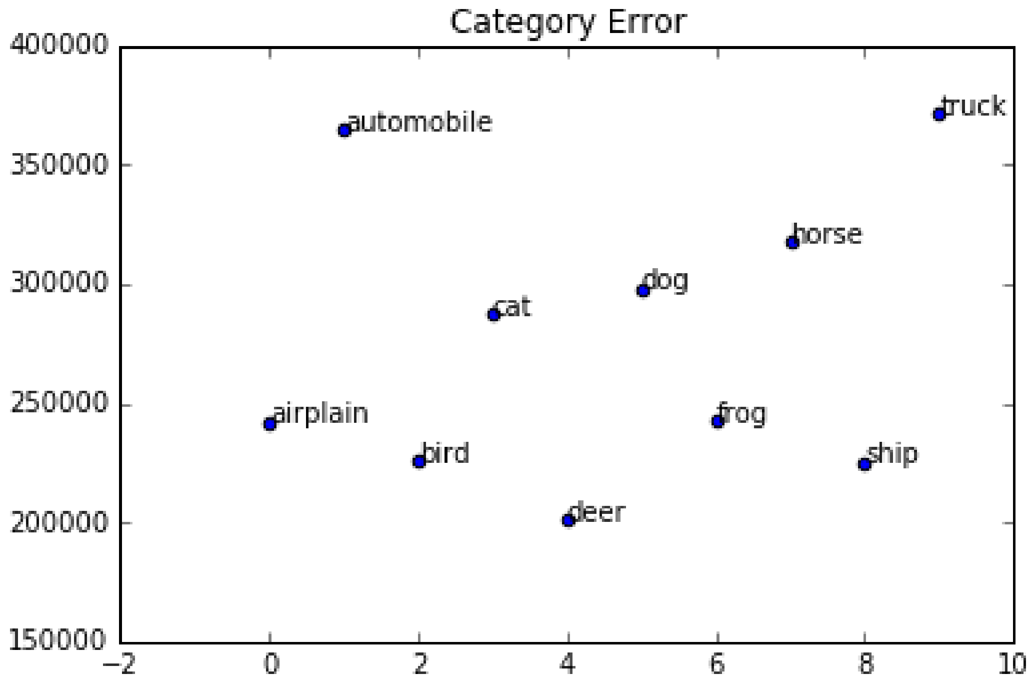

Figure 1: “Error against the Category”

(b)

For this question, we want to compute the distances between mean images for each pair of classes. We used the following to compute

$$D_{ij}^{(2)} = (x_i - x_j)^T(x_i - x_j)$$

We formed a 10×10 matrix, the non-diagnoal elements are the distances among the categories. We have $C_2^{10} = 45$ elements of distances in the upper triangle and the lower triangle. The diagonal elements are all 0.

Firstly, we made a matrix **A**

$$\mathbf{A} = [I - \frac{1}{N}\mathbf{1}\mathbf{1}^T]$$

Then we formed a matrix W

$$W = \frac{1}{2}AD^{(2)}A^T$$

Then we found the eigenvectors U and diagnoal matrix Λ.

$$WU = U\Lambda$$

Since we wanteed a 2D map, we chose the r to be 2 as the dimensions we wish to represent. We formed $\Lambda_r^{1/2}$ whose entries are the positive square roots of Λ_r.

$$V^T = \Lambda_r^{1/2}U_r^T = [v_1, \ldots, v_n]$$

Our distance matrix is below(scaled)

```
                           ##Distance Matrix##
array([[ 0.        ,  6.60249158,  6.29421314,  7.47268731,  8.42652319,
         7.70675095,  9.59090089,  6.5241017 ,  3.70800433,  5.68272516],
       [ 6.60249158,  0.        ,  3.47543823,  4.02999915,  4.48267203,
         4.76893893,  4.67134117,  3.72857285,  5.11163352,  3.72547361],
       [ 6.29421314,  3.47543823,  0.        ,  2.02867256,  2.35784445,
         2.75085579,  3.58332359,  1.64029924,  6.10868634,  5.55558719],
       [ 7.47268731,  4.02999915,  2.02867256,  0.        ,  1.84232045,
         1.6163987 ,  2.65683125,  2.3387323 ,  7.25966471,  6.57438375],
       [ 8.42652319,  4.48267203,  2.35784445,  1.84232045,  0.        ,
         2.42234173,  1.80592521,  2.68371337,  8.10047708,  7.17937588],
       [ 7.70675095,  4.76893893,  2.75085579,  1.6163987 ,  2.42234173,
         0.        ,  3.24933746,  3.308518  ,  7.44153655,  7.37350499],
       [ 9.59090089,  4.67134117,  3.58332359,  2.65683125,  1.80592521,
         3.24933746,  0.        ,  3.72040792,  8.82039131,  7.50290541],
       [ 6.5241017 ,  3.72857285,  1.64029924,  2.3387323 ,  2.68371337,
         3.308518  ,  3.72040792,  0.        ,  6.51085521,  5.28366315],
       [ 3.70800433,  5.11163352,  6.10868634,  7.25966471,  8.10047708,
         7.44153655,  8.82039131,  6.51085521,  0.        ,  4.18408483],
       [ 5.68272516,  3.72547361,  5.55558719,  6.57438375,  7.17937588,
         7.37350499,  7.50290541,  5.28366315,  4.18408483,  0.        ]])
```

Then we follow the procedure 4.3 to do the principal coordinate analysis. The 2D map is below

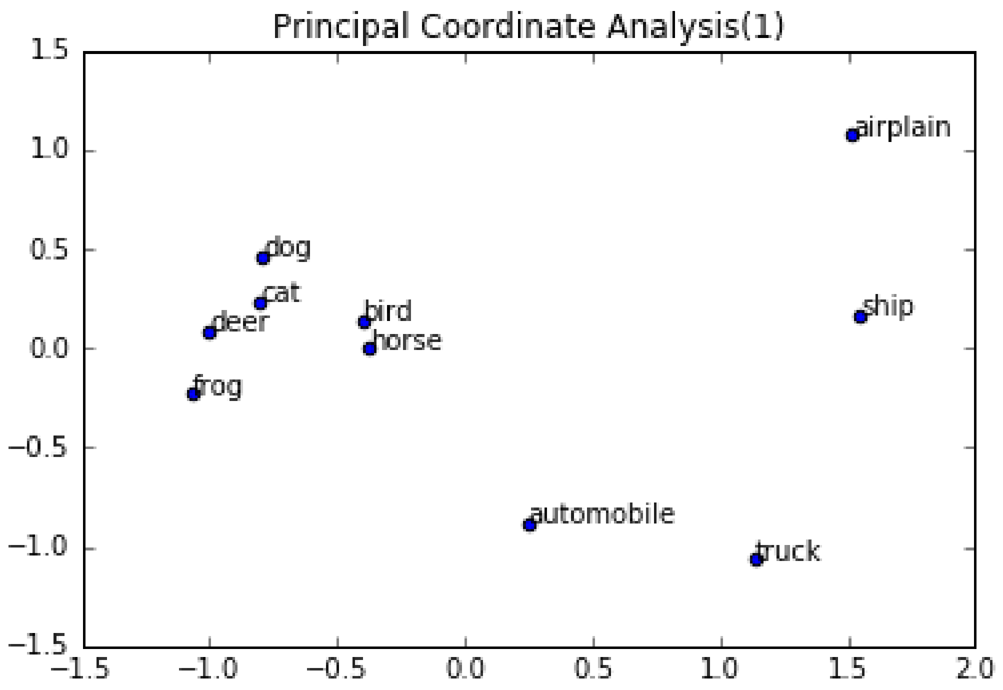

Figure 2: 2D map of the means of each category

(c)

We first calculated the error $E(A \to B), E(A \to C), \ldots$. It means we use the mean of class A and the first 20 principal components of class B. Then we calculated the error $E(B \to A), E(C \to A), \ldots$. We first calculated the error $E(A \to B), E(A \to C), \ldots$. It means we use the mean of class B and the first 20 principal components of class A. We got the similarity by $1/2(E(A \to B) + E(B \to A))$.

```
                          ##similarity between classes##
array([[      0.        ,  564725.8457277 ,  506325.3062958 ,
         584887.30503054,  584264.44844314,  605982.3875949 ,
         666015.79259579,  563982.494652  ,  415760.42884754,
         541044.93310706],
       [ 564725.8457277 ,       0.        ,  367668.03873925,
         418067.6409045 ,  384513.23174471,  442750.67602601,
         413650.56733711,  432902.62048656,  425571.53163891,
         463710.38954453],
       [ 506325.3062958 ,  367668.03873925,       0.        ,
         281402.49210339,  251734.70843938,  300114.20762   ,
         291344.98184875,  304727.74787942,  369475.80312319,
         421482.87359878],
       [ 584887.30503054,  418067.6409045 ,  281402.49210339,
              0.        ,  264741.80209499,  324596.01875686,
         303560.47343741,  341781.51366155,  435834.71805675,
         488150.09846431],
       [ 584264.44844314,  384513.23174471,  251734.70843938,
         264741.80209499,       0.        ,  284872.15924506,
         256115.69064922,  311494.37956466,  451546.82682342,
         483590.80610241],
       [ 605982.3875949 ,  442750.67602601,  300114.20762   ,
         324596.01875686,  284872.15924506,       0.        ,
         333769.34530657,  391353.96082297,  477976.19788537,
         550003.11406939],
       [ 666015.79259579,  413650.56733711,  291344.98184875,
         303560.47343741,  256115.69064922,  333769.34530657,
              0.        ,  363156.35497279,  525932.05175647,
         539187.79215019],
       [ 563982.494652  ,  432902.62048656,  304727.74787942,
         341781.51366155,  311494.37956466,  391353.96082297,
         363156.35497279,       0.        ,  525707.33227415,
         573119.37948911],
       [ 415760.42884754,  425571.53163891,  369475.80312319,
         435834.71805675,  451546.82682342,  477976.19788537,
         525932.05175647,  525707.33227415,       0.        ,
         403158.37573955],
       [ 541044.93310706,  463710.38954453,  421482.87359878,
         488150.09846431,  483590.80610241,  550003.11406939,
         539187.79215019,  573119.37948911,  403158.37573955,       0.        ]])
```

Comparing these two plots, we actually found they are pretty similar. We can see there are about 6 points clustered at left side. They are frog, dog, bird, cat, deer and horse and they are all animals. Also the relative positions of other four categories did not change much. The reason can be that no matter what measure we are using, the relative distance between points by the principal coordinate analysis is preserved.

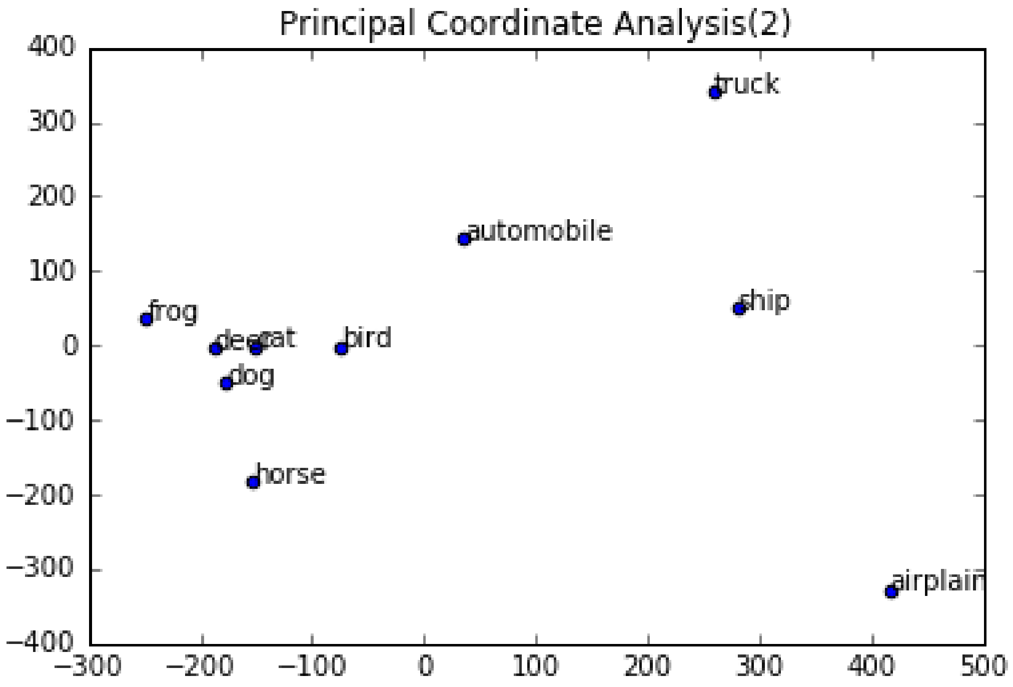

Figure 3: 2D map of the means of each category