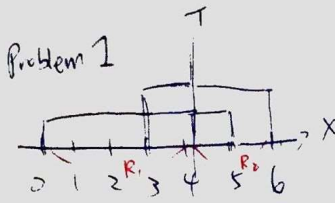


DLCV hw1 r06946003 湯忠憲

Problem1

Problem 1



$$P(x|w_1) = \begin{cases} \frac{1}{5}, & x \in [0, 5] \\ 0, & \text{otherwise} \end{cases}$$

$$P(x|w_2) = \begin{cases} \frac{1}{3}, & x \in [3, 6] \\ 0, & \text{otherwise} \end{cases}$$

Let T as our decision boundary.

$$P_e = \int_T^\infty P(x|w_1) P(w_1) dx + \int_{-\infty}^T P(x|w_2) P(w_2) dx$$

$$= \int_T^5 \frac{1}{5} \cdot \frac{3}{4} dx + \int_3^T \frac{1}{3} \cdot \frac{1}{4} dx = \frac{1}{2} - \frac{1}{15}T$$

$3 \leq T \leq 5 \Rightarrow T$ should be 5 for minimizing the error

$\Rightarrow P_e = \frac{1}{6} \Rightarrow$ decision Region ~~R_1 and R_2~~ $\xrightarrow{T=5}$

Problem2

In [124]:

```
from os import listdir
import imageio
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score
% matplotlib inline
```

In [118]:

```
# training & set name
train_set_name = [str(i)+"_"+str(j)+".png" for i in range(1,41) for j in range(1,7)]
test_set_name = [str(i)+"_"+str(j)+".png" for i in range(1,41) for j in range(7,11)]
```

In [119]:

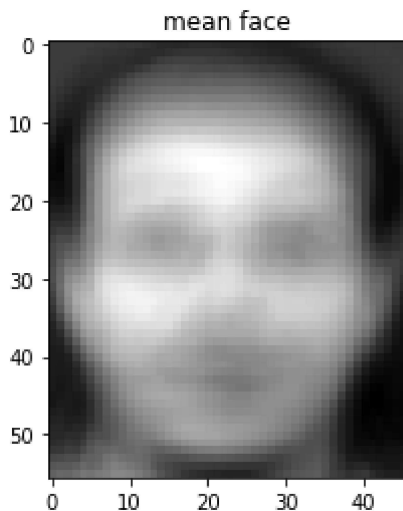
```
train_X = [imageio.imread("hw1_dataset/"+n) for n in train_set_name]
train_X = np.array(train_X).reshape(240,-1)
train_y = [i for i in range(1,41) for _ in range(1,7)]

test_X = [imageio.imread("hw1_dataset/"+n) for n in test_set_name]
test_X = np.array(test_X).reshape(160,-1)
test_y = [i for i in range(1,41) for _ in range(7,11)]
```

(a)

In [130]:

```
# mean face
mean_face_vector = train_X.mean(axis=0)
plt.title("mean face")
plt.imshow(mean_face_vector.reshape(56,46), cmap='gray')
plt.show()
```



In [44]:

```
# PCA and plot first three eigenfaces
pca = PCA()
output = pca.fit(train_X - mean_face_vector)
output.components_.shape
```

Out[44]:

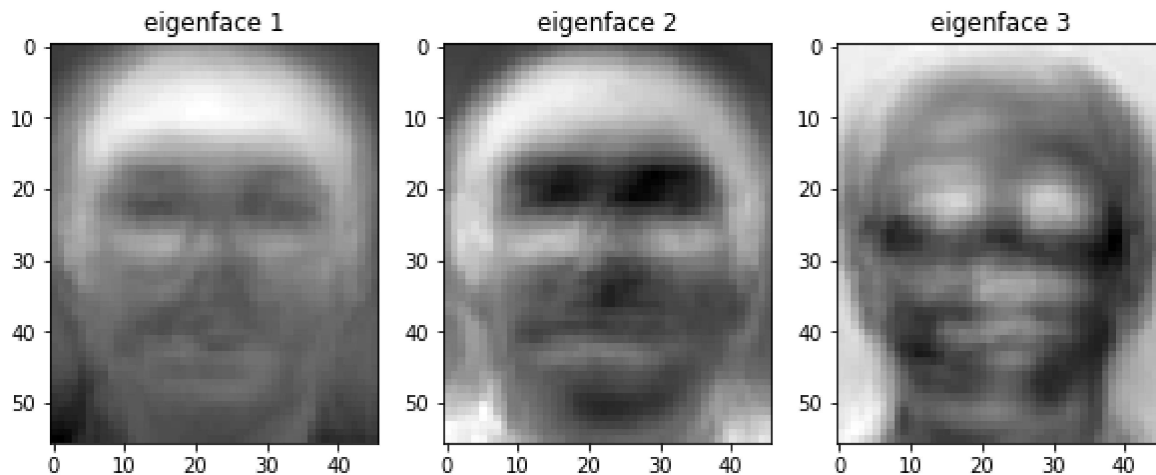
(240, 2576)

In [48]:

```
e1 = (output.components_[0]).reshape(56,46)
e2 = (output.components_[1]).reshape(56,46)
e3 = (output.components_[2]).reshape(56,46)
```

In [50]:

```
plt.figure(figsize=(10,6))
plt.subplot(131)
plt.imshow(e1,cmap='gray')
plt.title("eigenface 1")
plt.subplot(132)
plt.imshow(e2,cmap='gray')
plt.title("eigenface 2")
plt.subplot(133)
plt.imshow(e3,cmap='gray')
plt.title("eigenface 3")
plt.show()
```



(b)

In [131]:

```
input_img = imageio.imread("hw1_dataset/1_1.png").reshape(1,-1)
plt.title("original image")
plt.imshow(input_img.reshape(56,46), cmap="gray")
plt.show()
```



In [103]:

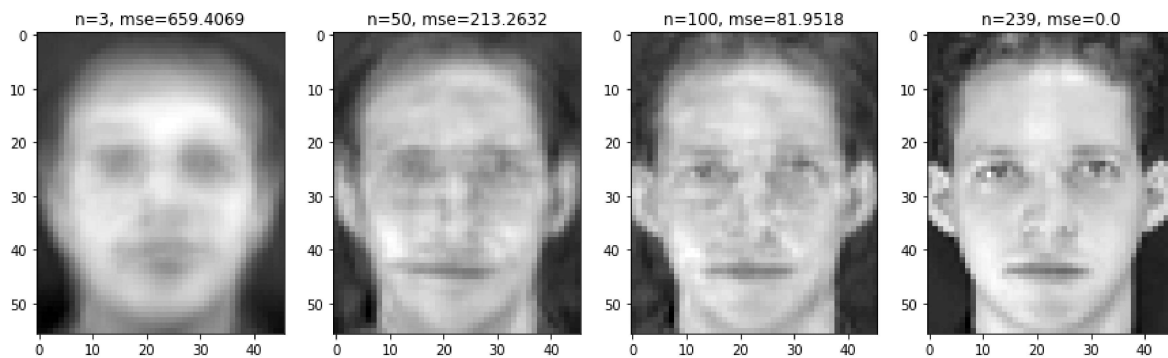
```
projected = pca.transform(input_img - mean_face_vector)
print(projected.shape)
```

(1, 240)

In [93]:

```
plt.figure(figsize=(15,6))

for j,i in enumerate([ 3, 50, 100, 239]):
    recon_f = (projected[:, :i] @ output.components_[ :i]) + mean_face_vector
    mse = np.mean((recon_f - input_img)**2)
    plt.subplot(1,4,j+1)
    tit = "n="+str(i)+", mse="+str(np.round(mse,4))
    plt.title(tit)
    plt.imshow(recon_f.reshape(56,46), cmap = "gray")
plt.show()
```



(c)

In [146]:

```
train_X_reduced = pca.transform(train_X - mean_face_vector)
knn = KNeighborsClassifier()
param_grid = {"n_neighbors": [1, 3, 5]}
clf = GridSearchCV(knn, param_grid, cv=3)
print("      k=1      k=3      k=5")
for n in [3, 50, 159]:
    clf.fit(train_X_reduced[:, :n], train_y)
    print("n= %3d" %n, clf.cv_results_["mean_test_score"])
```

	k=1	k=3	k=5
n= 3	[0.70833333	0.5875	0.4875
n= 50	[0.92916667	0.875	0.775
n= 159	[0.925	0.87083333	0.74583333]

From the above gridsearch output, we can find out the best parameters should be k=1 and n=50.

In [125]:

```
# real test
test_X_reduced = pca.transform(test_X - mean_face_vector)
knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(train_X_reduced[:, :50], train_y)
pred_y = knn.predict(test_X_reduced[:, :50])
```

In [128]:

```
acc = accuracy_score(y_pred=pred_y, y_true=test_y)
print("acc on testing set:", acc)
```

acc on testing set: 0.9625

Bonux

In []: