CSE315: Assignment #2

Due on Sunday, January 6, 2019

Bowen Du 11:59pm

Yuexuan Li 1507955

Question 1

Task1 Classification

The program run for task 1 is 'Task1.py'.

(a) Using MNIST database, Implement two classification algorithms and achieve a high accuracy (more than 90%). (20 ')

Classifier 1: KNN

sklearn.neighbor.KNeighborsClassifier is used to train the MNIST data. The **n_neighbors** is set as 3. Use **tensorflow.examples.tutorials.mnist.input_data** to obtain and load the MNIST data. Use **.score()** to compute the accuracy. The codes can be seen in the figure below:

```
print("classifier 1, KNN starts")
kn = KNeighborsClassifier(n_neighbors=3)
kn.fit(mnist.train.images, mnist.train.labels)

score = kn.score(mnist.test.images, mnist.test.labels)

print(" KNN, score: {:.6f}".format(score))
```

The screenshot below shows the accuracy of KNN is 0.968600

```
Please use alternatives such as official/mnist/dataset.py fro
classifier 1, KNN starts
 KNN, score: 0.968600
classifier 1, Convolutional neural network starts
WARNING:tensorflow:From /Users/effy/PycharmProjects/assignmer
Instructions for updating:
```

Classifier 2: CNN

The codes for CNN model using **tensorflow** are shown in the screenshots below. Generally speaking, I use 2 **convolutional layers**, 2 **pooling layers** and 1 **fully connected layer**. Also, I use **dropout** after fully connected layer to prevent overfitting. Beyond that, add **softmax** in the output layer.

Question 1 continued on next page...

```python
def conv2d(input, filter):
    return tf.nn.conv2d(input, filter, strides=[1, 1, 1, 1], padding='SAME')


def max_pool(input):
    return tf.nn.max_pool(input, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='SAME')


# initialize weight
def weight_variable(shape):
    initial = tf.truncated_normal(shape, stddev=0.1)
    return tf.Variable(initial)

# initialize bias
def bias_variable(shape):
    return tf.Variable(tf.zeros(shape))
```

Question 1 continued on next page...

```
print("classifier 1, Convolutional neural network starts")

input = tf.placeholder(tf.float32, [None, 784])
input_image = tf.reshape(input, [-1, 28, 28, 1])

y = tf.placeholder(tf.float32, [None, 10])

# convolutional layer 1
W1 = weight_variable([5, 5, 1, 32])
b1 = bias_variable([32])
h_conv1 = tf.nn.relu(conv2d(input_image, W1) + b1)
# pooling layer 1
h_pool1 = max_pool(h_conv1)

# convolutional layer 2
W2 = weight_variable([5, 5, 32, 64])
b2 = bias_variable([64])
h_conv2 = tf.nn.relu(conv2d(h_pool1, W2) + b2)
# pooling layer 2
h_pool2 = max_pool(h_conv2)

# fully connected layer
W_fc1 = weight_variable([7 * 7 * 64, 1024])
b_fc1 = bias_variable([1024])
h_pool2_flat = tf.reshape(h_pool2, [-1, 7 * 7 * 64])
h_fc1 = tf.nn.relu(tf.matmul(h_pool2_flat, W_fc1) + b_fc1)

# Dropout, prevent overfitting
keep_prob = tf.placeholder(tf.float32)
h_fc1_drop = tf.nn.dropout(h_fc1, keep_prob)

# output layer
W_fc2 = weight_variable([1024, 10])
b_fc2 = bias_variable([10])
y_conv = tf.matmul(h_fc1_drop, W_fc2) + b_fc2
cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(labels=y, logits=y_conv))

# training
train_step = tf.train.AdamOptimizer(1e-4).minimize(cross_entropy)
correct_prediction = tf.equal(tf.argmax(y_conv, 1), tf.argmax(y, 1))
accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))

with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    for i in range(10000):
        batch = mnist.train.next_batch(50)
        if i % 1000 == 0:
            train_accuracy = accuracy.eval(feed_dict={
                input: batch[0], y: batch[1], keep_prob: 1.0})
            print('step %d, training accuracy %g' % (i, train_accuracy))
        train_step.run(feed_dict={input: batch[0], y: batch[1], keep_prob: 0.5})

    print('Convolutional neural network, test accuracy %g' % accuracy.eval(
        feed_dict={input: mnist.test.images, y: mnist.test.labels, keep_prob: 1.0}))
```

The accuracy result for CNN is 0.9914

Question 1 continued on next page...

```
classifier 1, Convolutional neural network starts
WARNING:tensorflow:From /Users/effy/PycharmProjects/assign
Instructions for updating:

Future major versions of TensorFlow will allow gradients t
into the labels input on backprop by default.

See `tf.nn.softmax_cross_entropy_with_logits_v2`.

2018-12-15 12:23:59.671246: I tensorflow/core/platform/cpu
step 0, training accuracy 0.16
step 1000, training accuracy 0.98
step 2000, training accuracy 0.96
step 3000, training accuracy 1
step 4000, training accuracy 1
step 5000, training accuracy 1
step 6000, training accuracy 1
step 7000, training accuracy 1
step 8000, training accuracy 0.98
step 9000, training accuracy 1
Convolutional neural network, test accuracy 0.9914

Process finished with exit code 0
```

(b) Describe the techniques, including data preparation, feature reduction and training tricks in your classification algorithms. (10')

Classifier 1: KNN

Data Preparation: Just use the original data set provided by tensorflow, consisting of 55000 training data and 10000 testing data. Each data consist of image pixel matrix (28*28) and corresponding label.
Feature Reduction: None.
Training Tricks: Use Euclidian distance to measure how similar and dissimilar of the digits bitmaps are over raw pixels. By experience, the number of neighbors in the KNN model is 3. So every digit may obtain 3 neighbors.

Classifier 2: CNN

Data Preparation: As same as the KNN
Feature Reduction:
convolutional: stride 1, padding 0
pooling: max pool with 2*2 filters and stride 2
First Convolutional Layer:

Question 1 continued on next page...

1. Convolutional layer: weight_variable([5, 5, 1, 32]), bias_variable([32]), reshape, Relu Activation Function
2. Pooling Layer
Second Convolutional Layer:
1. Convolutional Layer: weight_variable([5, 5, 32, 64]), bias_variable([64]), reshape, Relu Activation Function
2. Pooling Layer
Fully Connected Layer: weight_variable([7 * 7 * 64, 1024]), bias_variable([1024]), reshape, Relu Activation Function
Dropout Layer: prevent overfitting
Output Layer: softmax
Training Tricks: AdamOptimizer, learning rate 0.0001, 10000 iterations, Cross Entropy Cost Function

(c) Analyse some other techniques can be applied in your classification algorithms to improve your model's performance (accuracy, efficiency and storage). (5')

Accuracy:
Checking if the classification algorithms are overfitting which might influence the accuracy. Thus, it had better use cross-validation to diagnose these two models which did not adopt it in the experiments. Also, as my KNN model does not use any feature reduction method, CNN can be used to extract image features before using KNN to learn the data.
Data augmentation will increase the accuracy to extent if used. This method makes slight change in the training data to increase the volume of training data so that the model robustness will be enhanced. We can use flip, random crop, color jittering and so on.
Efficiency:
The training complexity of my KNN model is O(d) where d is the number of dimensionality. The testing complexity is O(nd) where n is the number of training instances and d is the number dimensionality. Therefore, to make the KNN faster, it is necessary to reduce d the data dimensionality by e.g. feature selection or reduce n the number of training instances by not comparing all the training examples. Three strategies can be chosen to avoiding comparing every instance which are K-D trees, inverted lists and locality-sensitive hashing.
Storage: KNN is considered a lazy learning algorithm which greater storage requirements and higher computational costs on recall. CNN is computationally demanding as well because of the huge data size. One alternative is to used distributed machine learning which move the data on the cloud to have scalable storage. Google Cloud Platform provides the cloud computing service. The implementation can be obtained by tensorflow or apache spark.

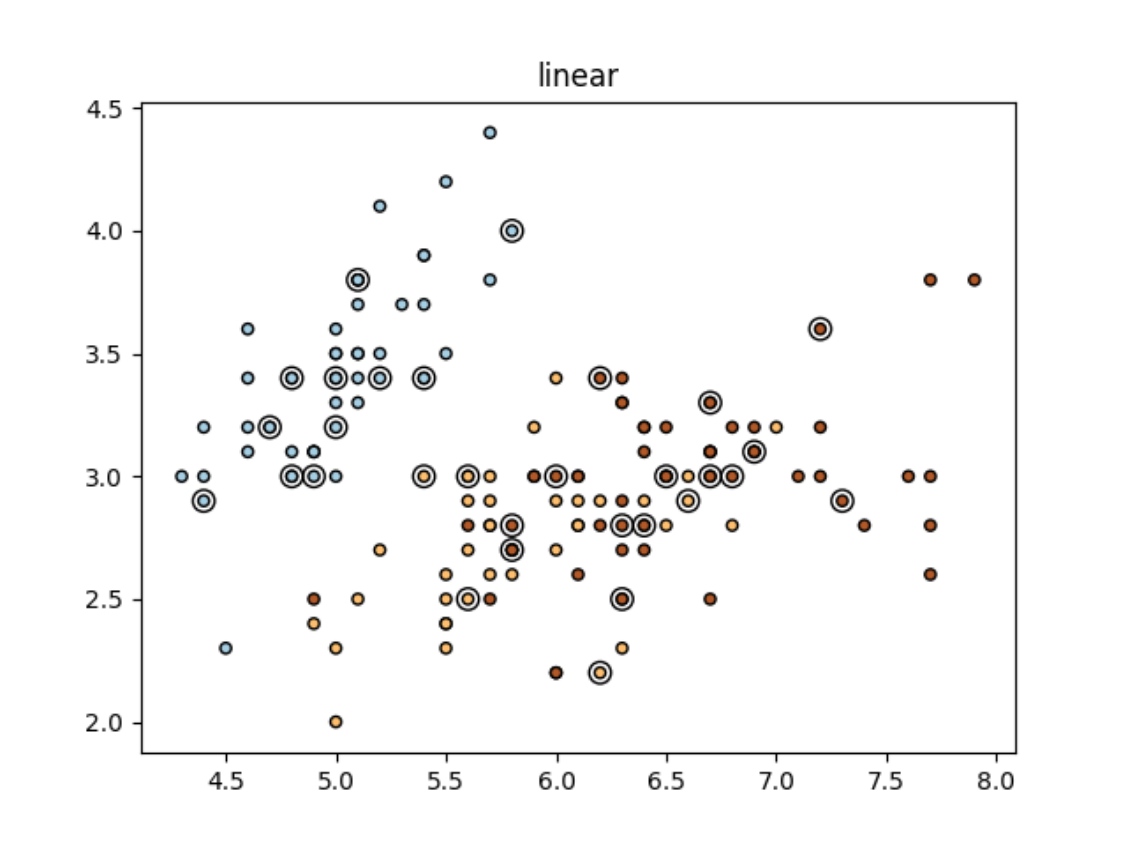
Question 2

Task 2 SVM and PCA

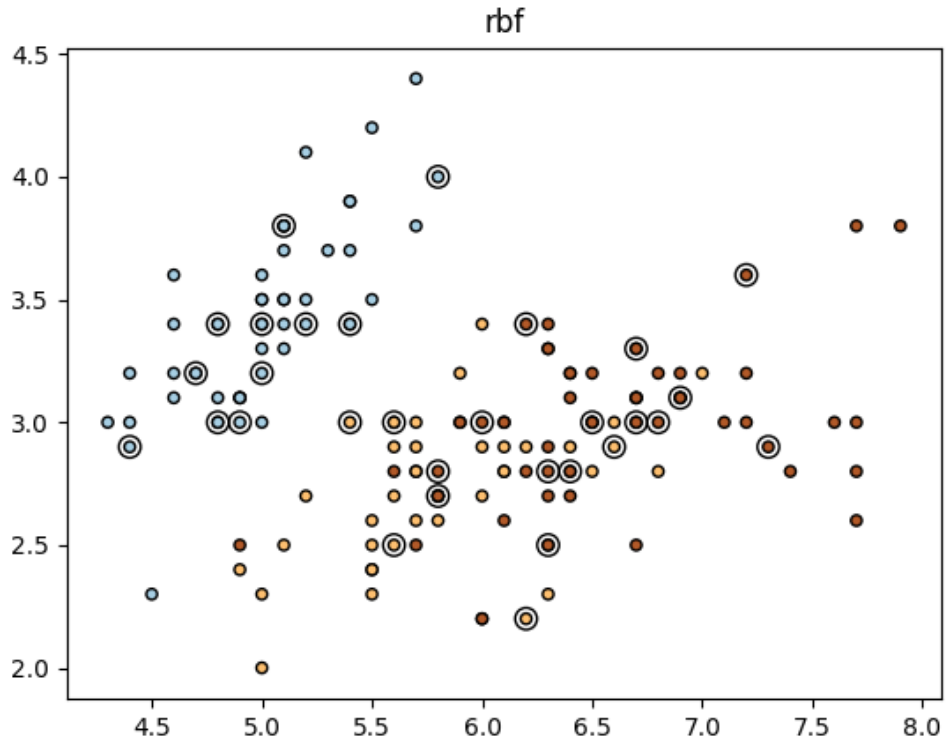
Question 2 continued on next page...

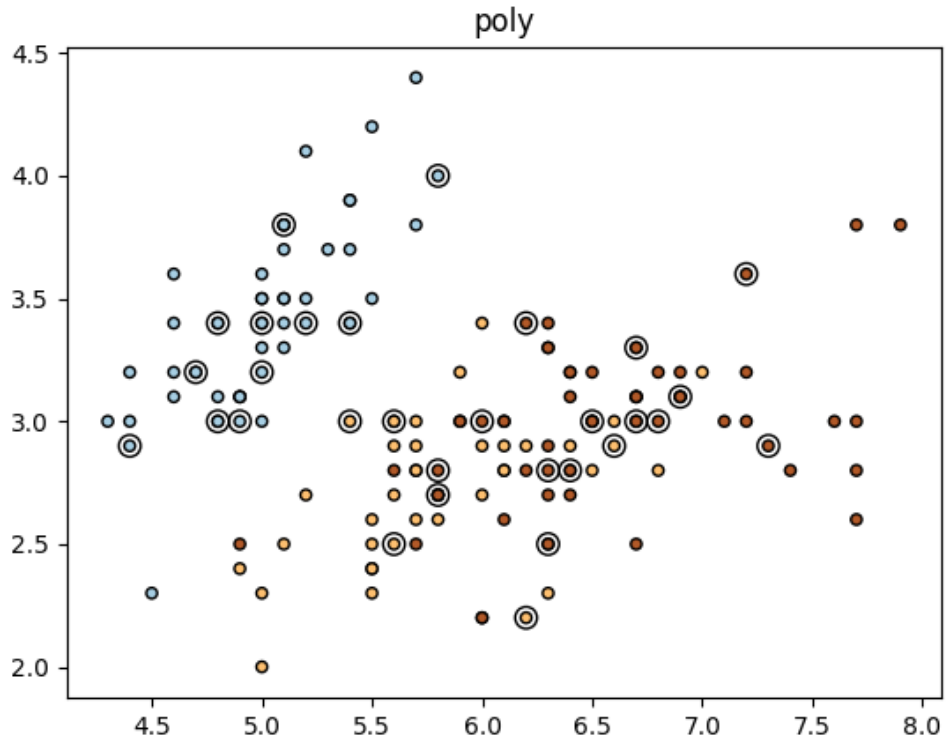
(a) Using iris.data, select training dataset and validation dataset and implement SVM algorithm (based on public packages or libraries) to classify the type of iris (achieving 90% accuracy). (5')

sklearn.svm is used to implement svm. 3 types of svm are employed which are **linear**, **rbf** and **poly**. All of them use **gamma=10**. The accuracy is computed by **sklearn.matrics.accuracy_score**. The dataset is divided into training set and testing set by **sklearn.model_selection. train_test_split**. 80% of data are training set, while 20% of data are testing set. The classification reusts are shown in the following figures:



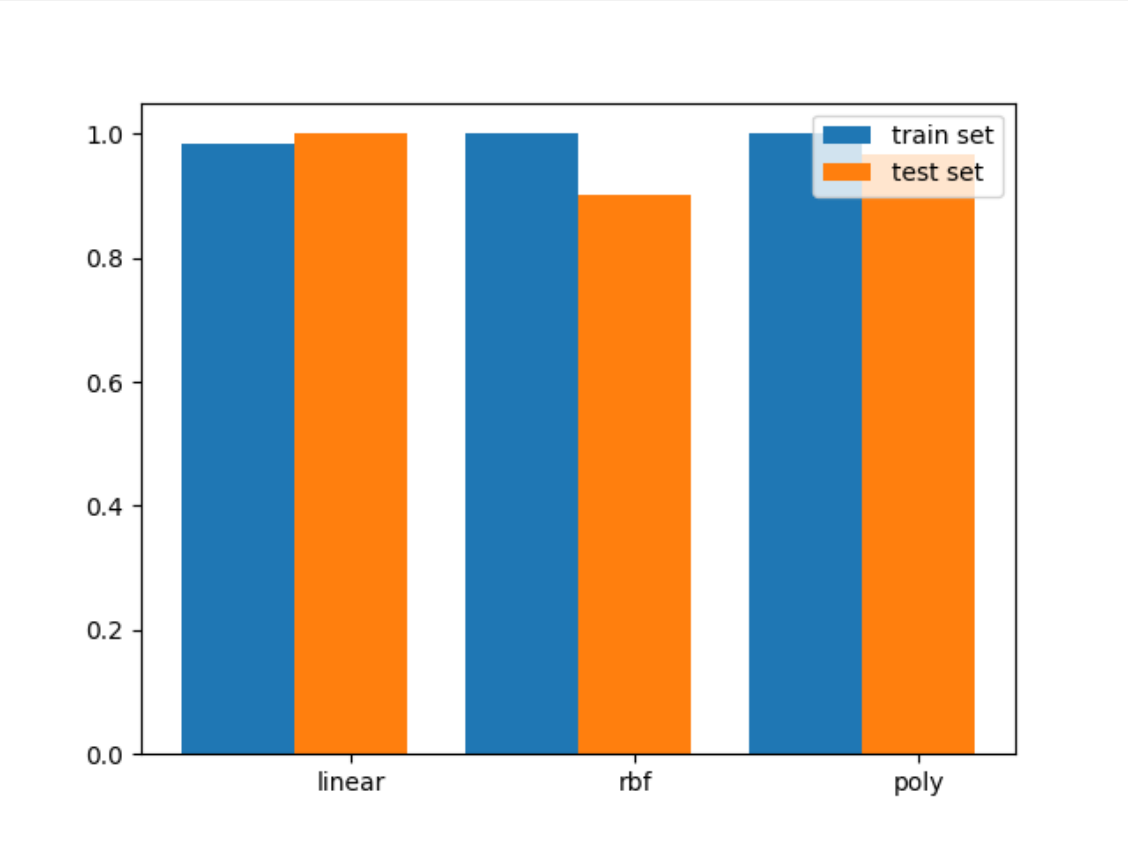
Question 2 continued on next page...





Question 2 continued on next page...

According to the figures belwo,all the models achieve more than 90% accuracy.




```
Train dataset, Accuracy ( linear ):  0.9833333333333333
Test dataset, Accuracy ( linear ):  1.0
Train dataset, Accuracy ( rbf ):  1.0
Test dataset, Accuracy ( rbf ):  0.9
Train dataset, Accuracy ( poly ):  1.0
Test dataset, Accuracy ( poly ):  0.9666666666666667
```

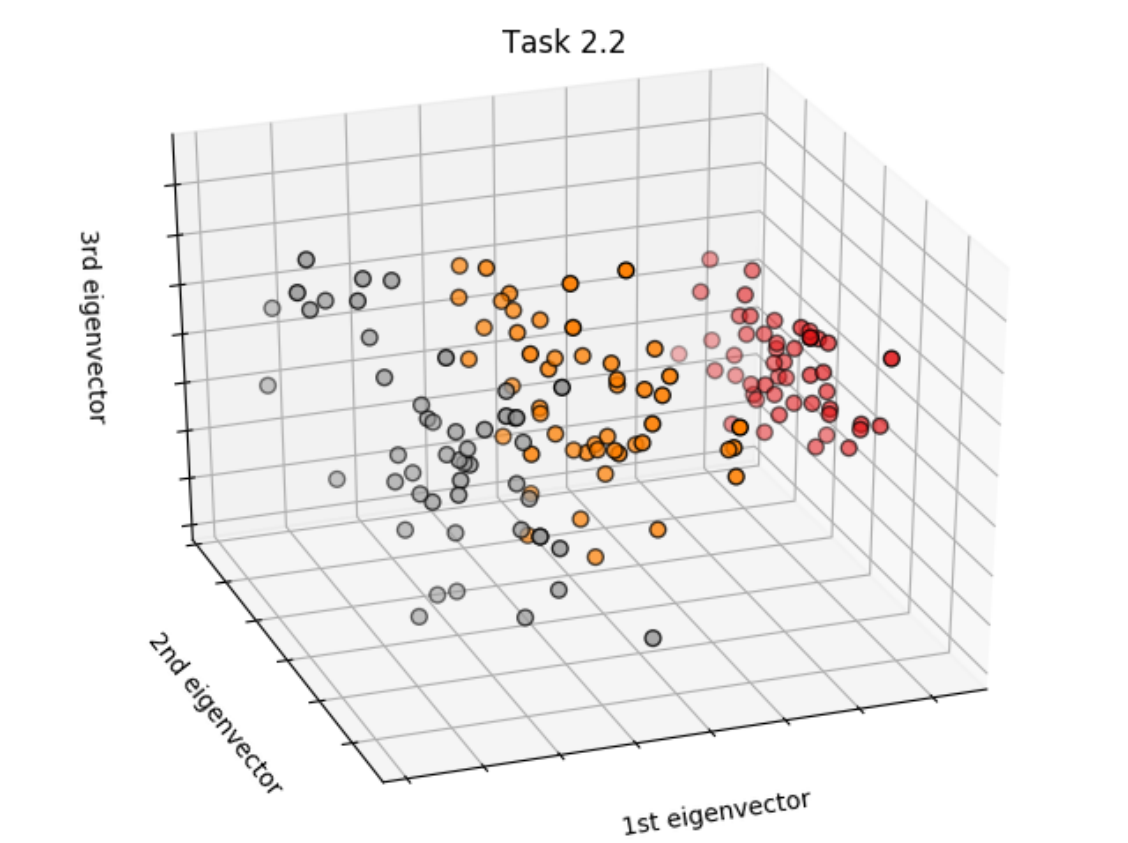
(b) Using iris.data, reduce the dimension of features and extract the first, second and third principal component. (5')

sklearn.decomposition.PCA is used to reduce the the dimension of features. **n_components=3** is adopted. The PCA variance ratio computed by **.explained_variance_ratio_.sum()** is shown as below:

```
PCA variance ratio 0.994816914549810 1
```

The reduced results are shown in the scatter graph below.

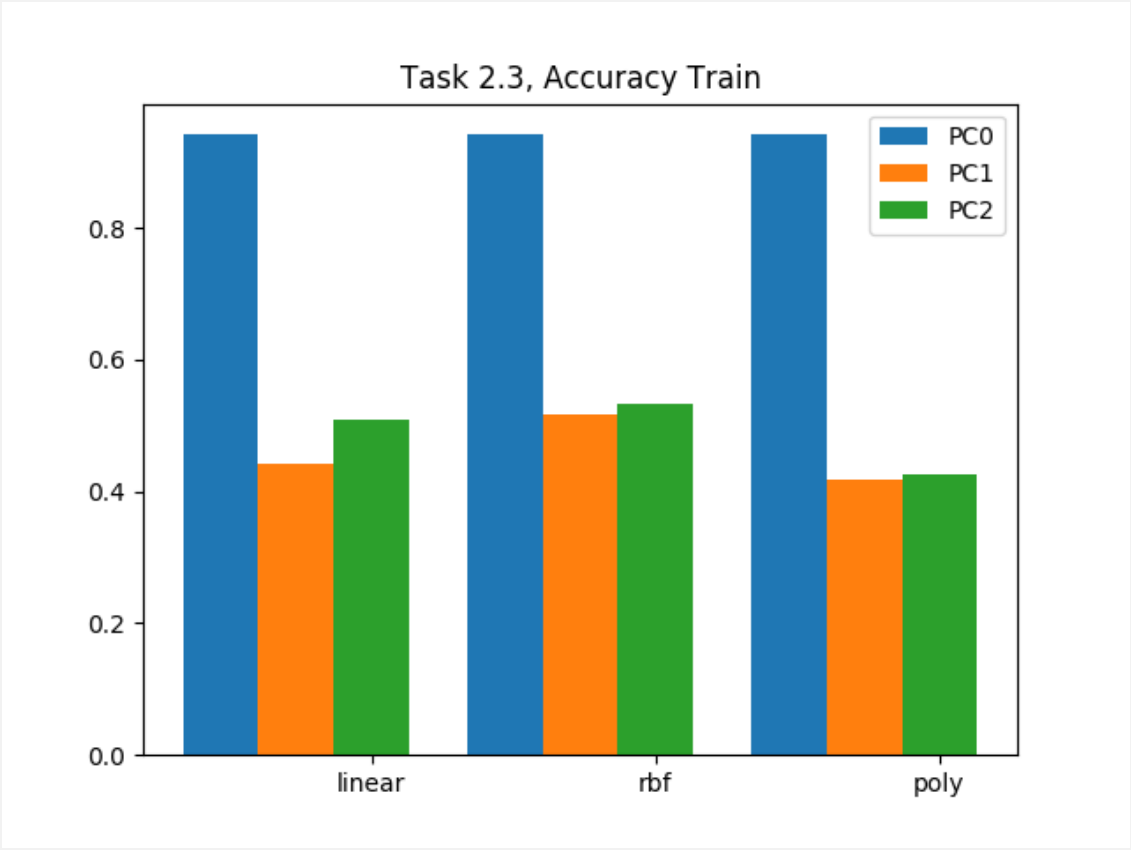
Question 2 continued on next page...

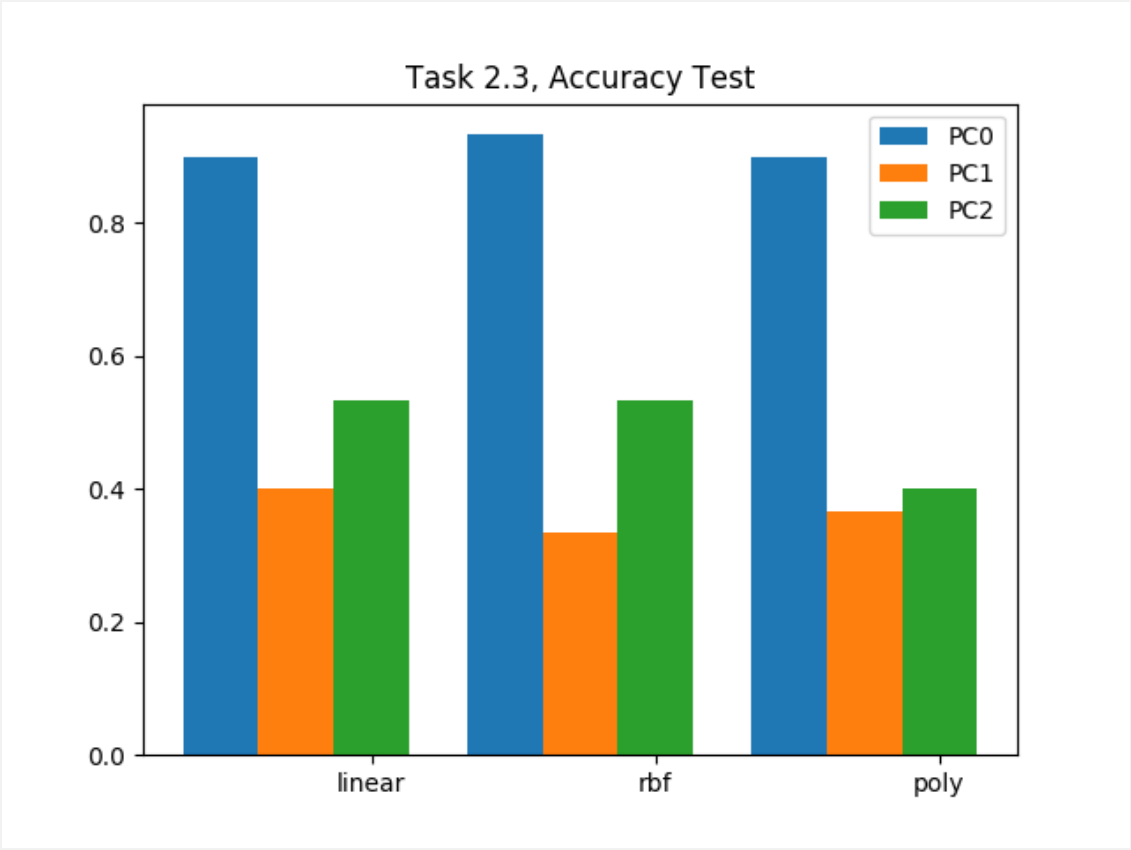


(c) Based on question 2, using the first, second and third principal component, respectively, to train a SVM model to classify the type of iris and compare their accuracy. (15')

sklearn.svm is used to implement svm. 3 types of svm are employed which are **linear**, **rbf** and **poly**. All of them use **gamma=10**. The accuracy is computed by **sklearn.matrics.accuracy_score**. The dataset is divided into training set and testing set by **sklearn.model_selection. train_test_split**. 80% of data are training set, while 20% of data are testing set. Since this task is to use principal components respectively to train svm, the classification results are not plotted. The accuracy results are shown in the figures below. As the figures are shown, the first principal component achieve the highest accuracy no matter what kind of svm is.

Question 2 continued on next page...



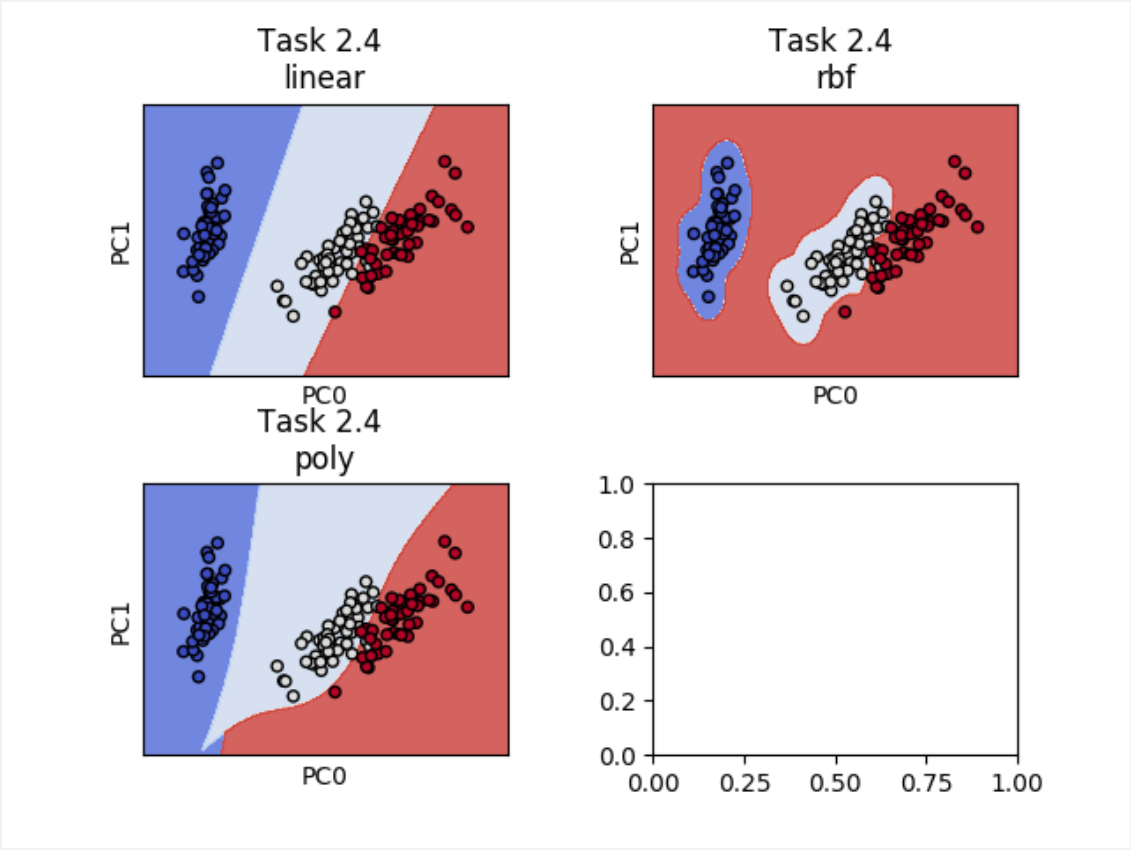


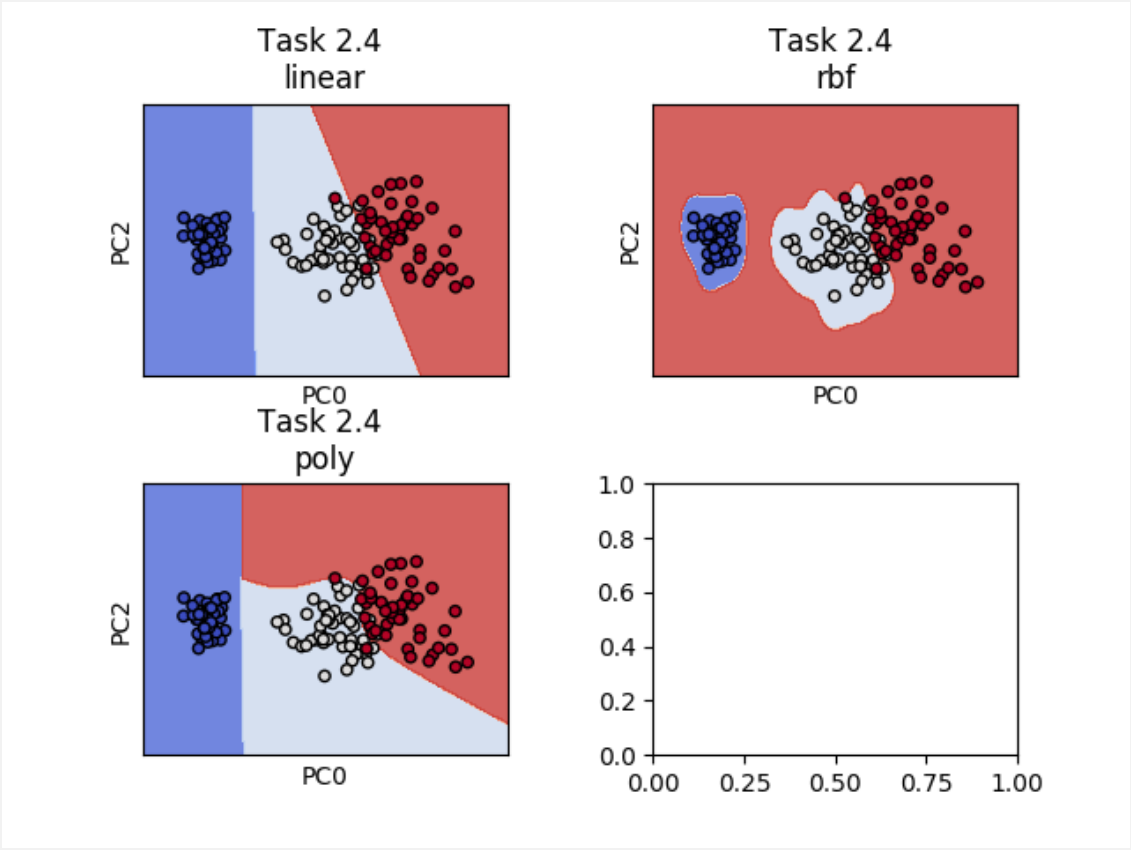
Question 2 continued on next page...

```
Task 2.3 .......
==============================
SVM, PC 0
Train dataset, Accuracy ( linear ):  0.9416666666666667
Test dataset, Accuracy ( linear ):  0.9
Train dataset, Accuracy ( rbf ):  0.9416666666666667
Test dataset, Accuracy ( rbf ):  0.9333333333333333
Train dataset, Accuracy ( poly ):  0.9416666666666667
Test dataset, Accuracy ( poly ):  0.9
==============================
SVM, PC 1
Train dataset, Accuracy ( linear ):  0.44166666666666665
Test dataset, Accuracy ( linear ):  0.4
Train dataset, Accuracy ( rbf ):  0.5166666666666667
Test dataset, Accuracy ( rbf ):  0.3333333333333333
Train dataset, Accuracy ( poly ):  0.4166666666666667
Test dataset, Accuracy ( poly ):  0.36666666666666664
==============================
SVM, PC 2
Train dataset, Accuracy ( linear ):  0.5083333333333333
Test dataset, Accuracy ( linear ):  0.5333333333333333
Train dataset, Accuracy ( rbf ):  0.5333333333333333
Test dataset, Accuracy ( rbf ):  0.5333333333333333
Train dataset, Accuracy ( poly ):  0.425
Test dataset, Accuracy ( poly ):  0.4
```

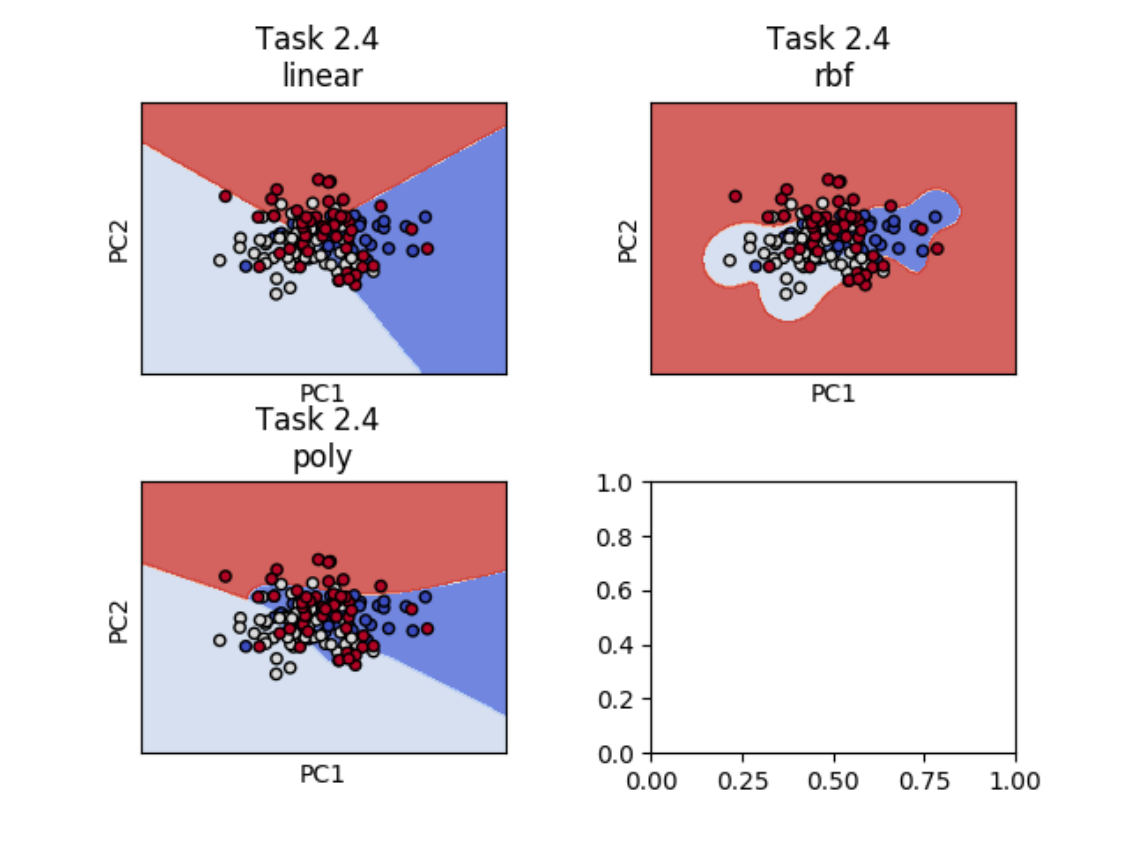
(d) Based on question 2,using any combination of the first, second and third principal components to train a SVM model to classify the type of iris and compare their accuracy. (15')

The methods in this task are the same as those in task 2.3. The classification results are shown as follows:

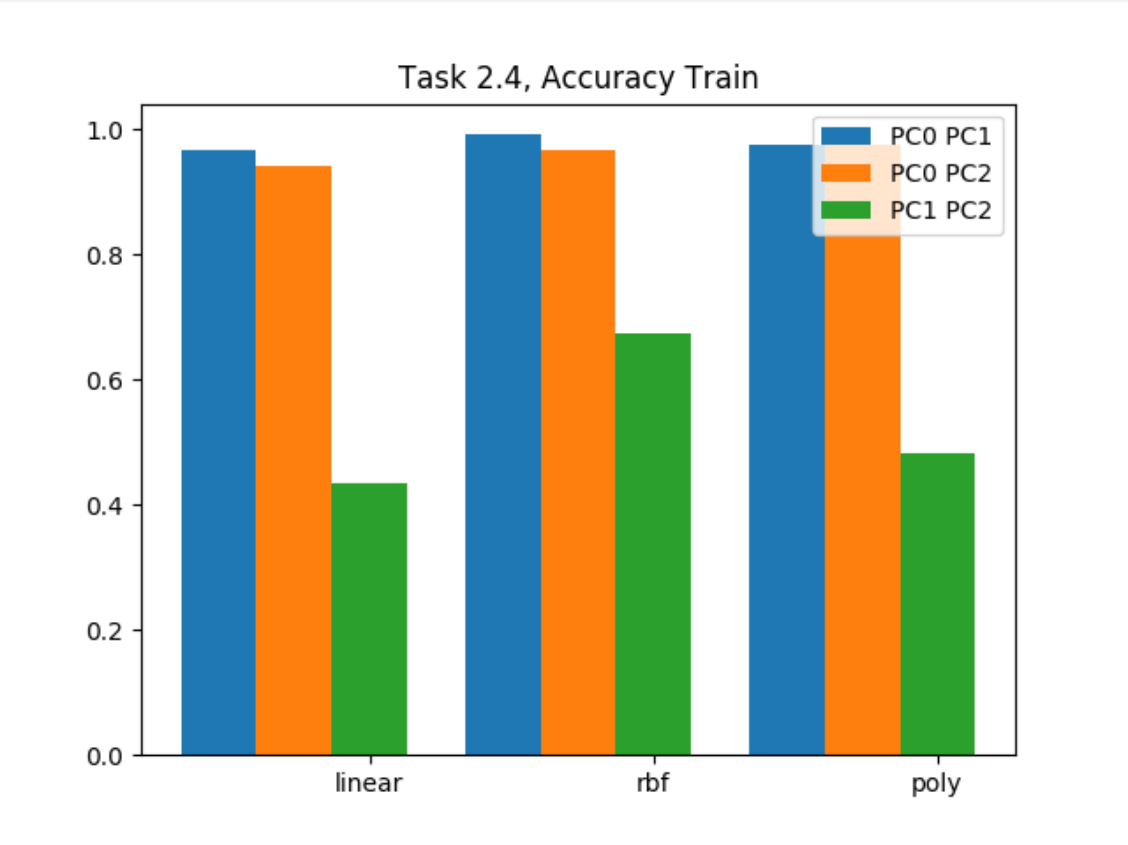


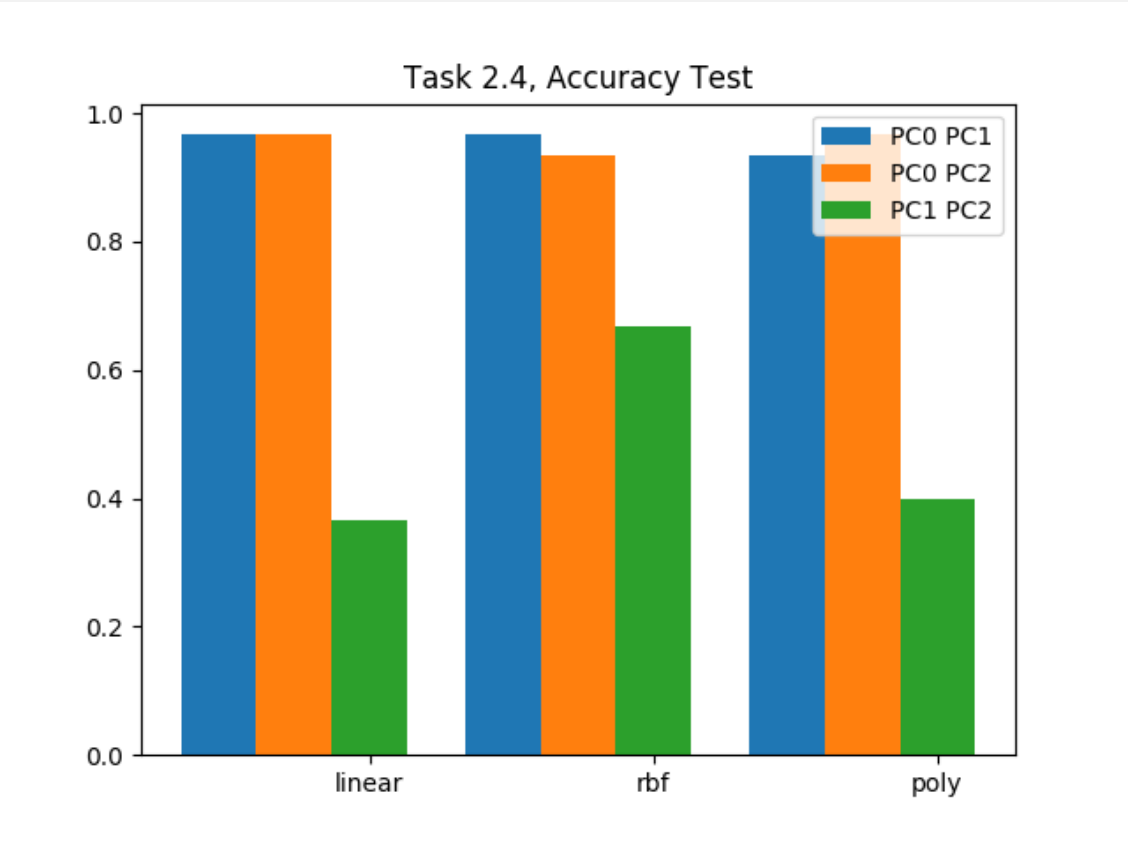


Question 2 continued on next page...



According to the figures below, it can be concluded that the combination of the first and the second principal components or the first and the third principal components give the better accuracy. The differences among liner, rbf and poly are very slight, which shows that the performance is mainly influenced by the combination of the principal components.





Question 2 continued on next page...

```
Task 2.4 .......
==============================
SVM, PC 0  PC1
Train dataset, Accuracy ( linear ):  0.9666666666666667
Test dataset, Accuracy ( linear ):  0.9666666666666667
Train dataset, Accuracy ( rbf ):  0.9916666666666667
Test dataset, Accuracy ( rbf ):  0.9666666666666667
Train dataset, Accuracy ( poly ):  0.975
Test dataset, Accuracy ( poly ):  0.9333333333333333
==============================
SVM, PC 0  PC2
Train dataset, Accuracy ( linear ):  0.9416666666666667
Test dataset, Accuracy ( linear ):  0.9666666666666667
Train dataset, Accuracy ( rbf ):  0.9666666666666667
Test dataset, Accuracy ( rbf ):  0.9333333333333333
Train dataset, Accuracy ( poly ):  0.975
Test dataset, Accuracy ( poly ):  0.9666666666666667
==============================
SVM, PC 1  PC2
Train dataset, Accuracy ( linear ):  0.43333333333333335
Test dataset, Accuracy ( linear ):  0.36666666666666664
Train dataset, Accuracy ( rbf ):  0.675
Test dataset, Accuracy ( rbf ):  0.6666666666666666
Train dataset, Accuracy ( poly ):  0.48333333333333334
Test dataset, Accuracy ( poly ):  0.4
```

Question 3

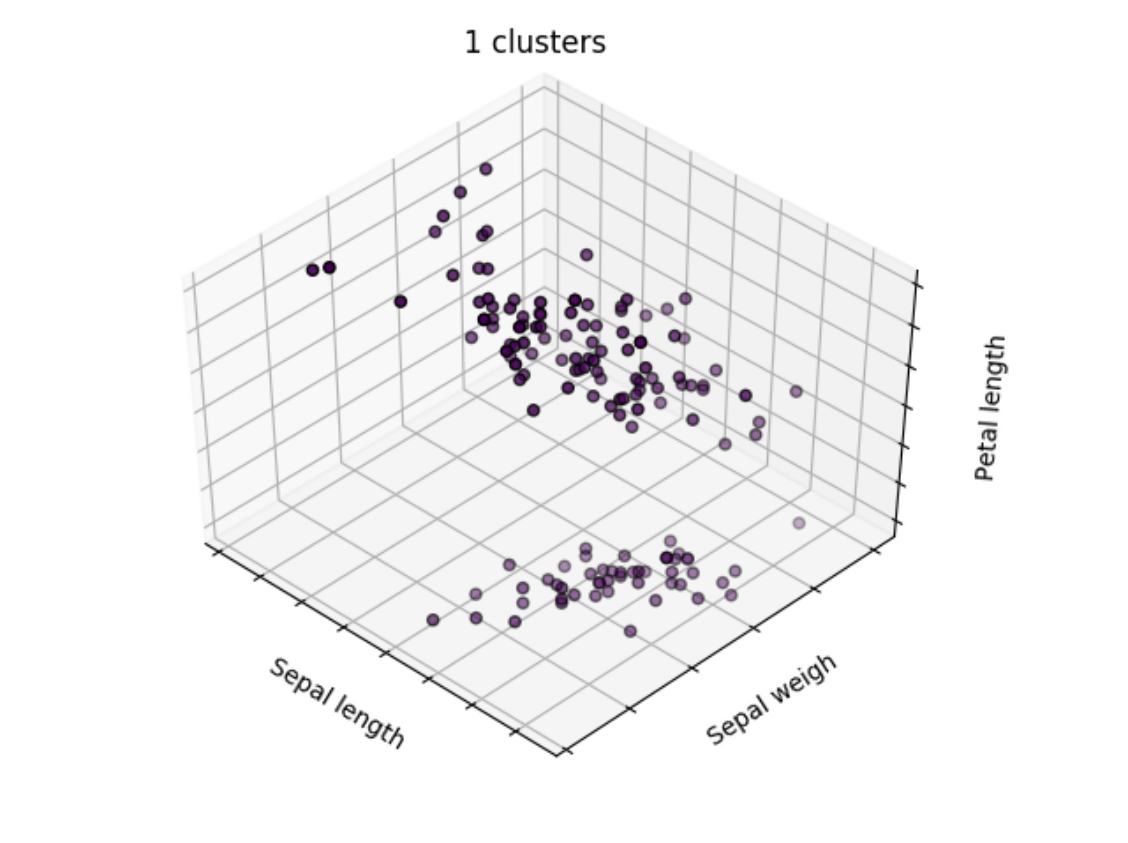
Task 3 Clustering

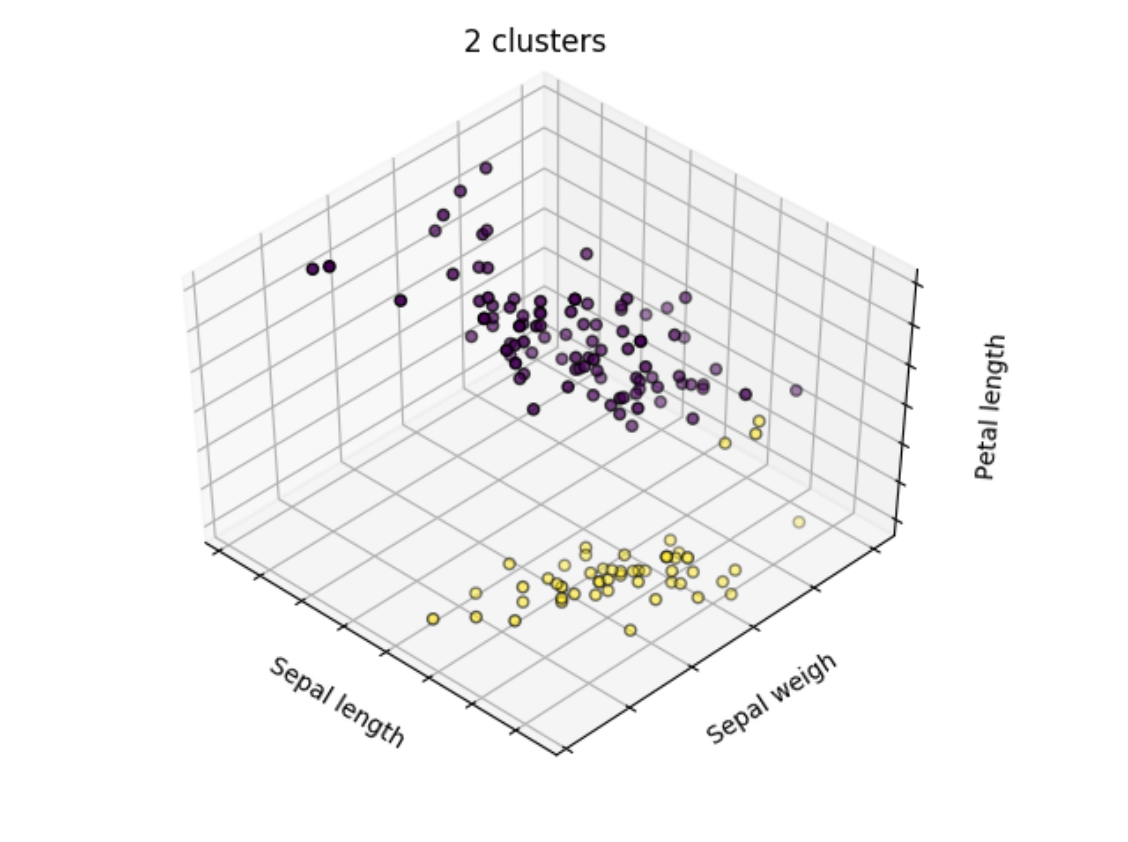
The program run for task 3.1 is 'Task3_1.py'. The program run for task 3.2 is 'Task3_2.py'. To implement k-means, **sklearn.cluster.KMeans** is used in these two tasks. To measure k-means performance, **sklearn.metrics.cluster.v_measure_score** and **sklearn.metrics.accuracy_score** are used in these two tasks.

(a) Using iris.data, select the number (1, 2, 3, 4, 5 and 6) of clustering and implement k-means algorithm (based on public packages or libraries). (5')

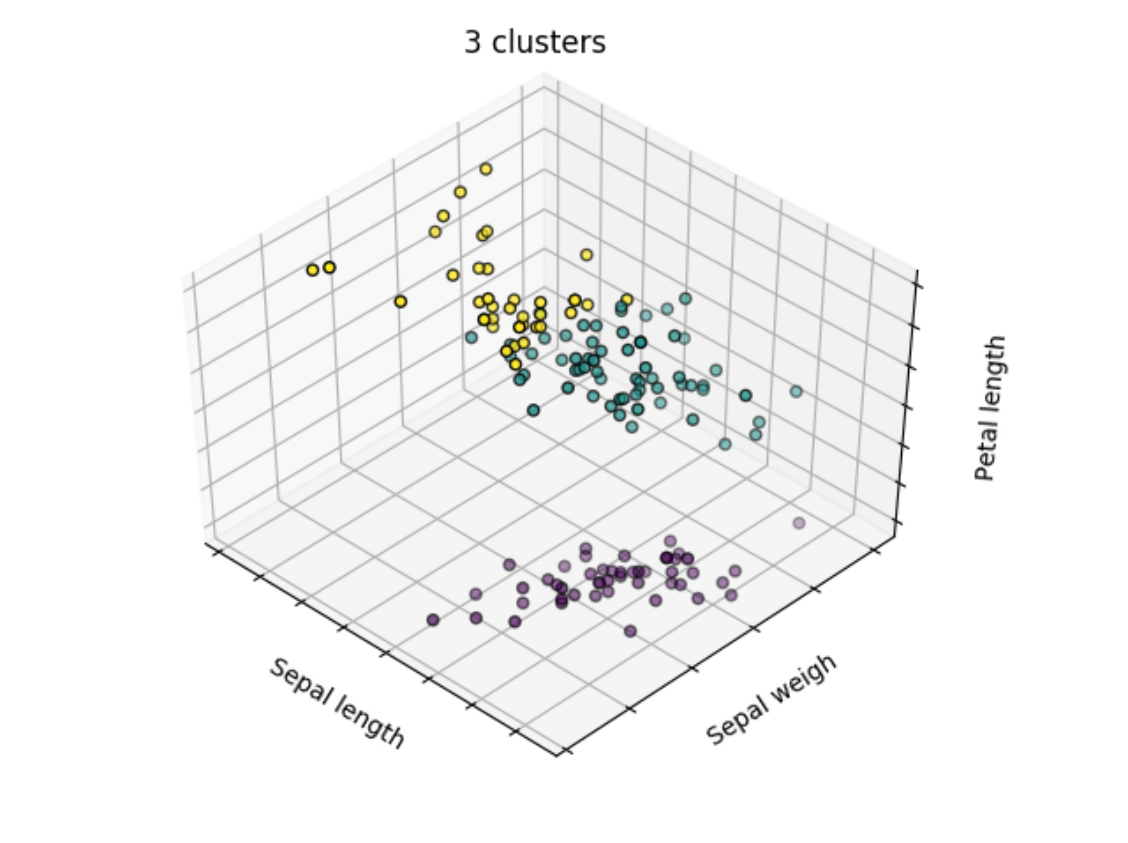
The classification results are shown as follows:

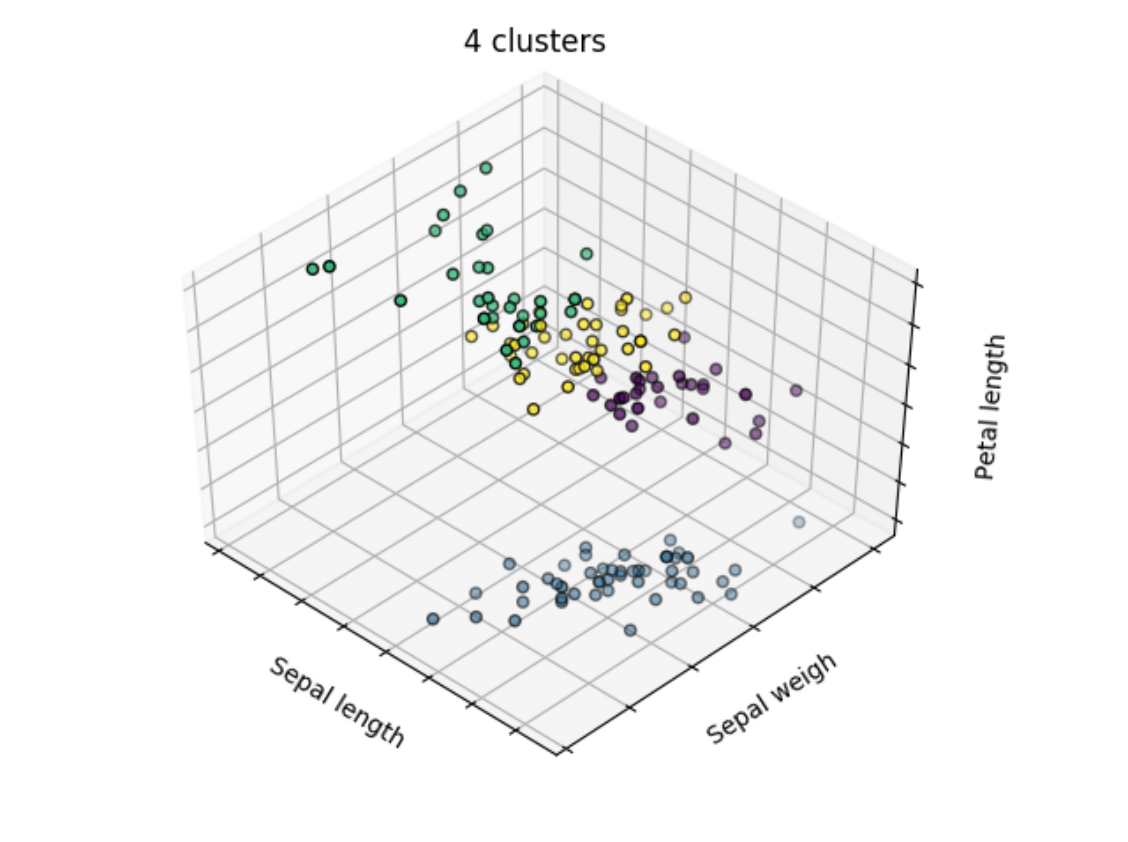
Question 3 continued on next page...



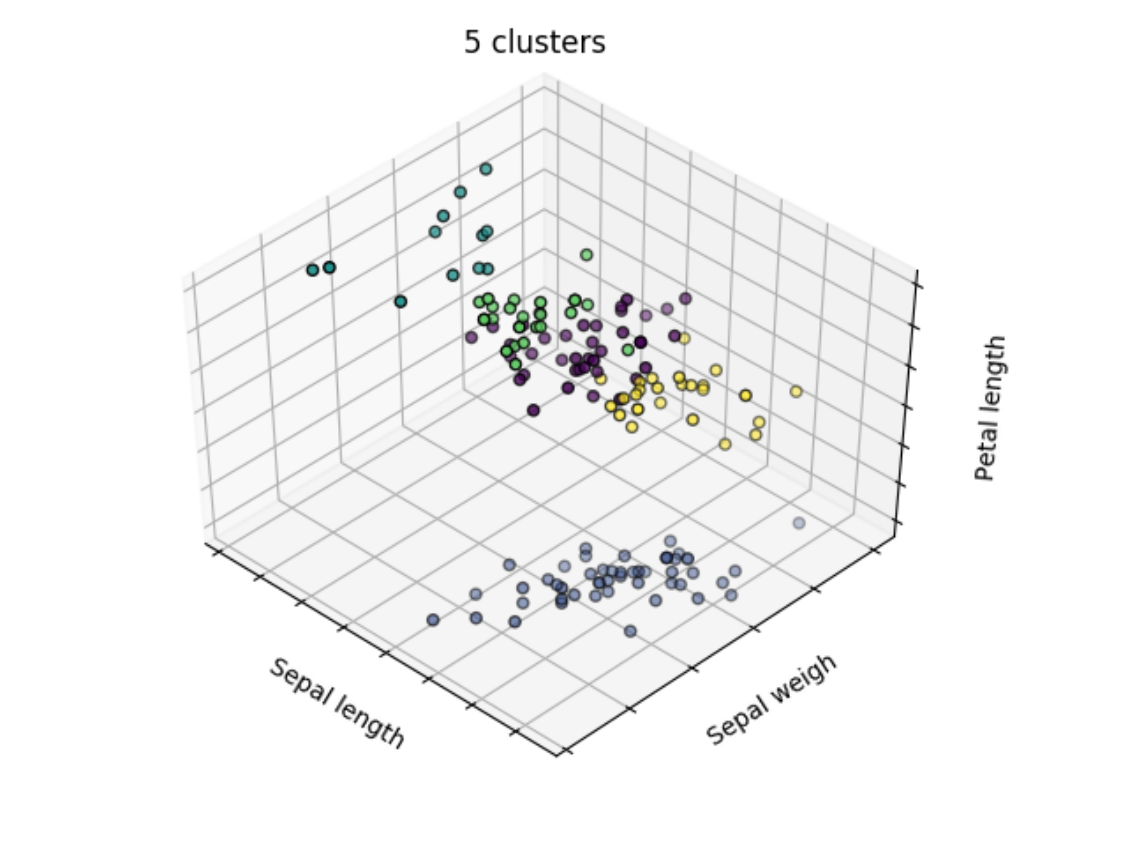


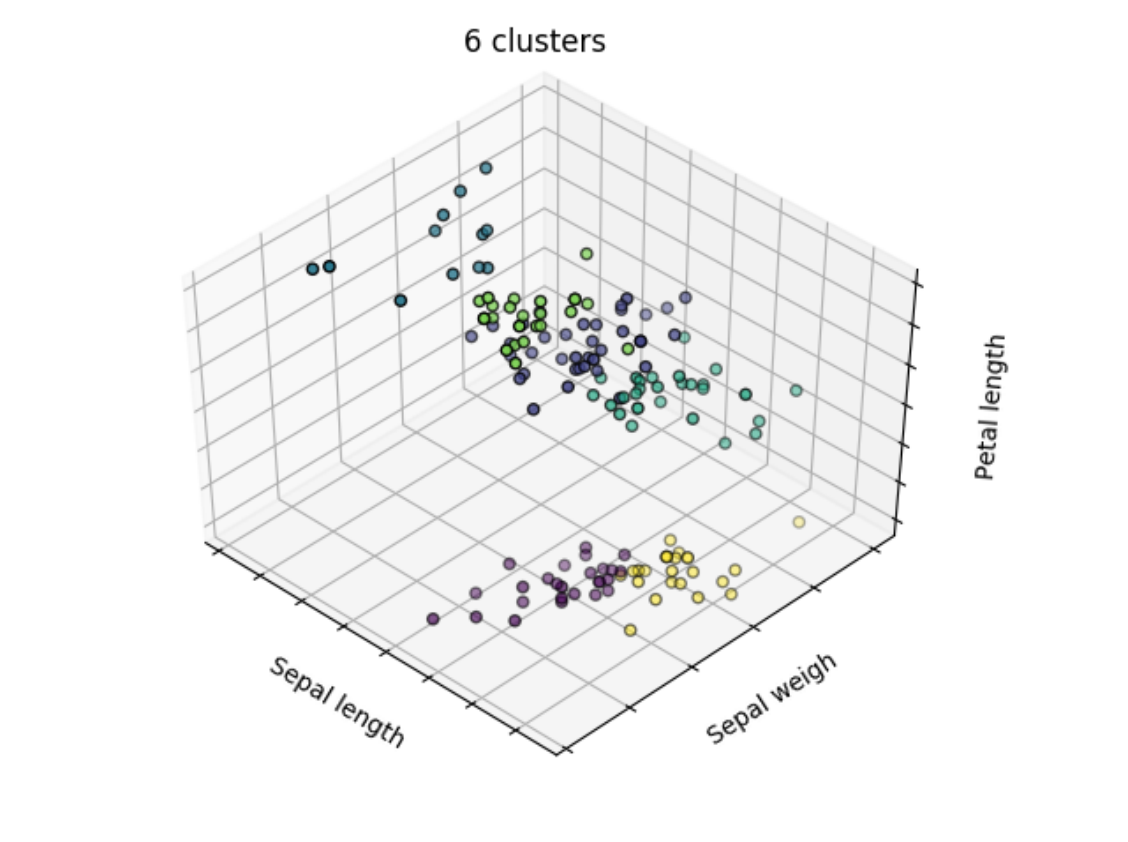
Question 3 continued on next page...





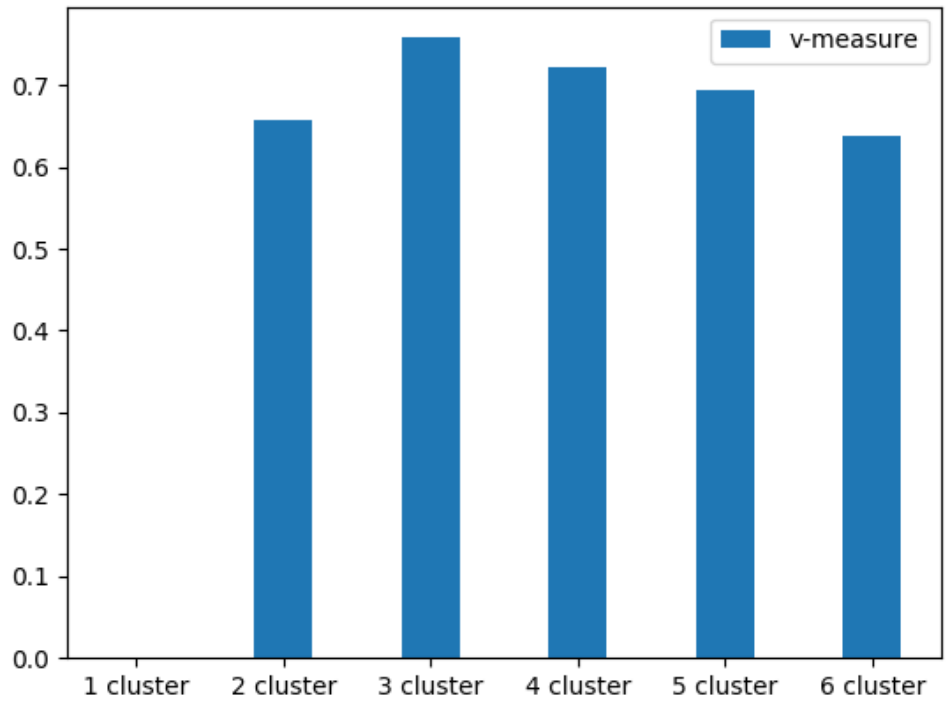
Question 3 continued on next page...





Question 3 continued on next page...

To evaluate k-means clustering performance, v-measure is used instead of accuracy. The reason why accuracy is not computed is that k-means is an unsupervised learning and the labels are unknown so that accuracy is not suitable to this task. V-measure provides an elegant solution to accurately evaluating and combing two desirable aspects of clustering which are homogeneity and completeness. The v-measure results of models with 1-6 clusters are shown in the figures below. The accuracy of 3 clusters is 0.24. 3-cluster show the best result.

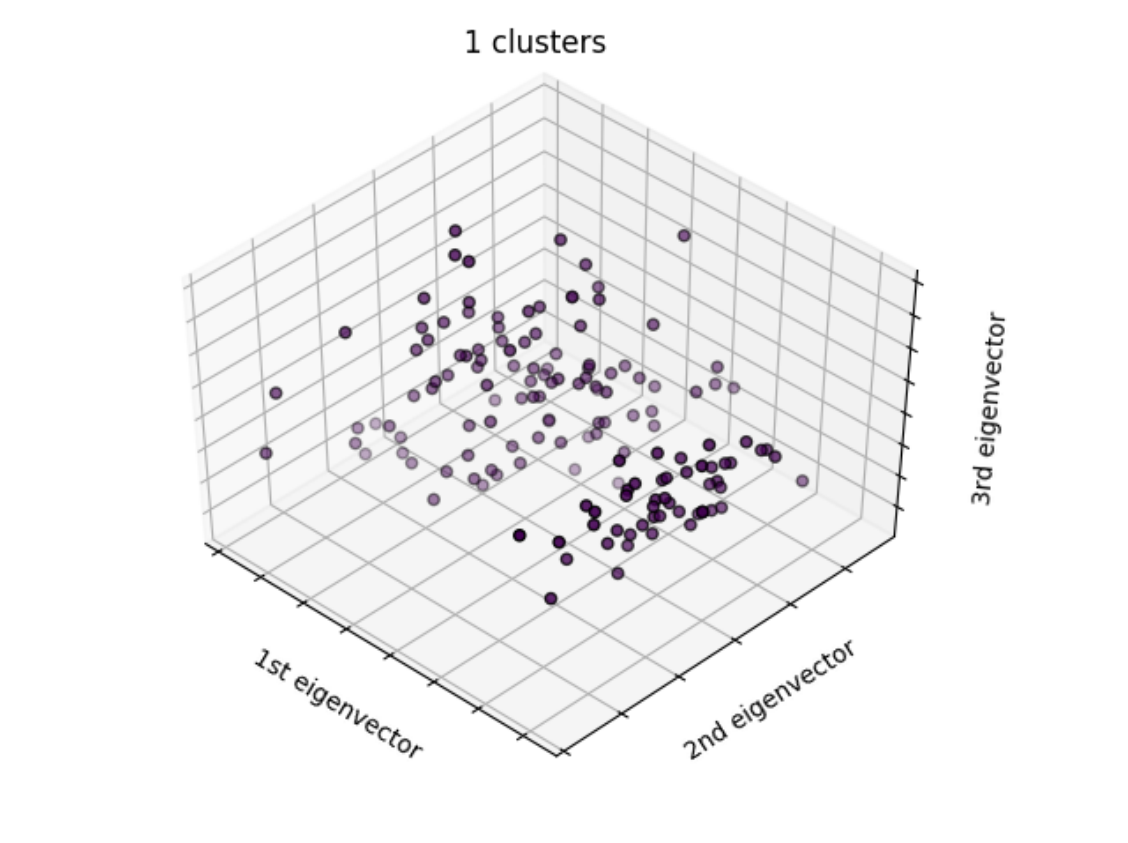


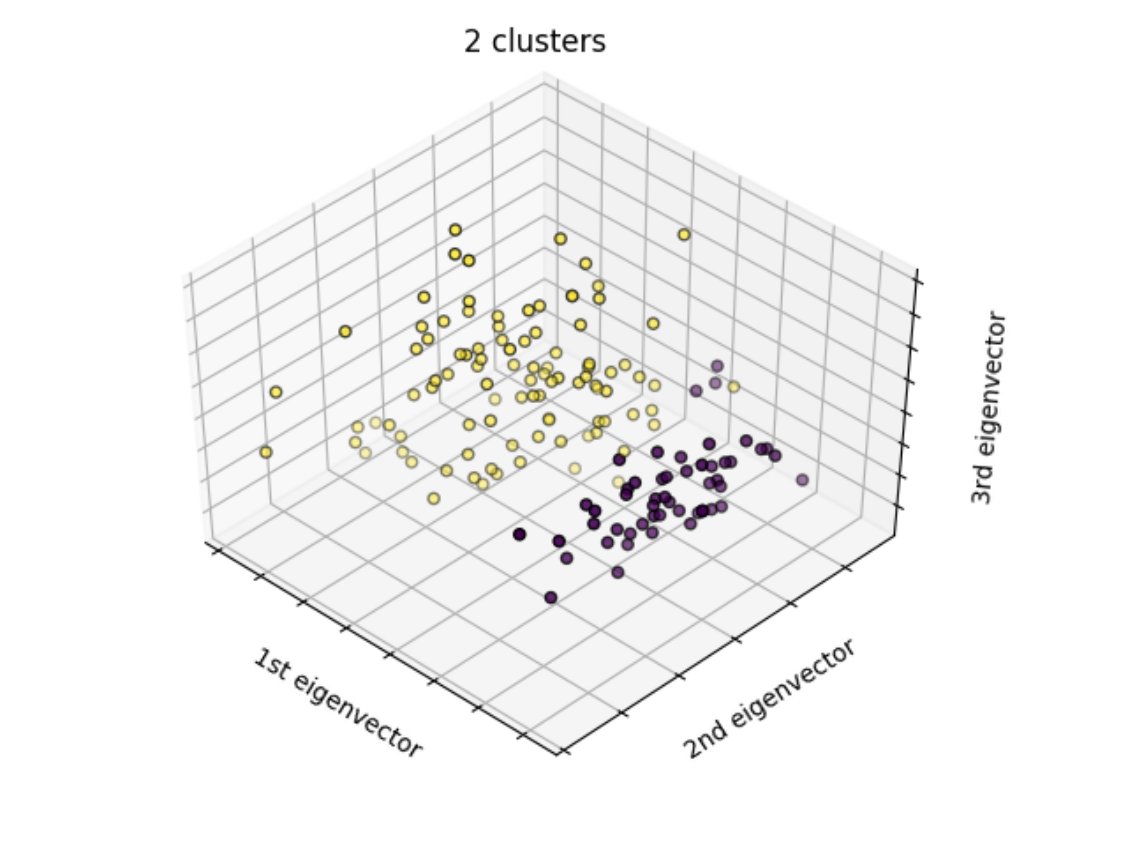
Question 3 continued on next page...

```
v measure ( k-means 1 clusters ):  -9.095116925863002e-16

v measure ( k-means 2 clusters ):  0.6565191143081123

v measure ( k-means 3 clusters ):  0.7581756800057786

v measure ( k-means 4 clusters ):  0.7140752149705604

v measure ( k-means 5 clusters ):  0.6938628345588717

v measure ( k-means 6 clusters ):  0.641542853905476

The accuracy of 3 clusters:  0.24
```

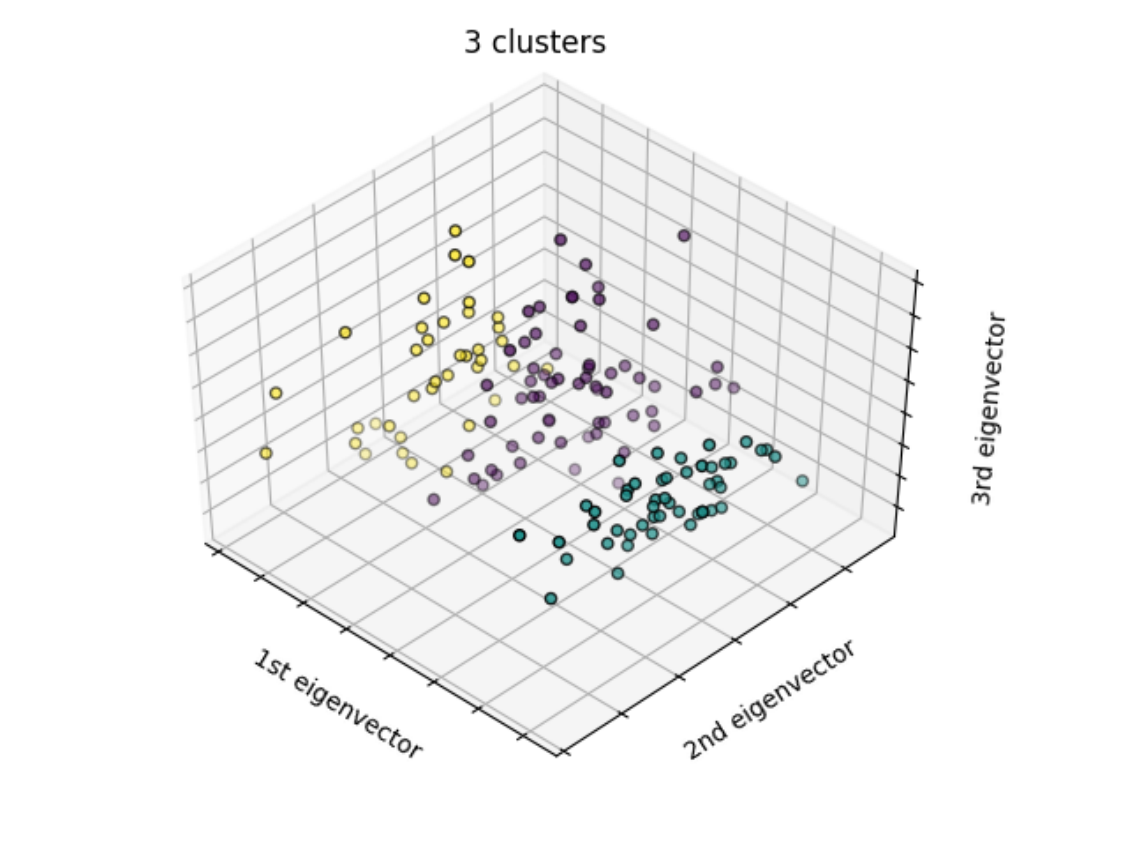
(b) Using PCA to reduce the dimension of features and combine the first, second and third principal components to implement k-means algorithm (based on public packages or libraries). (10')

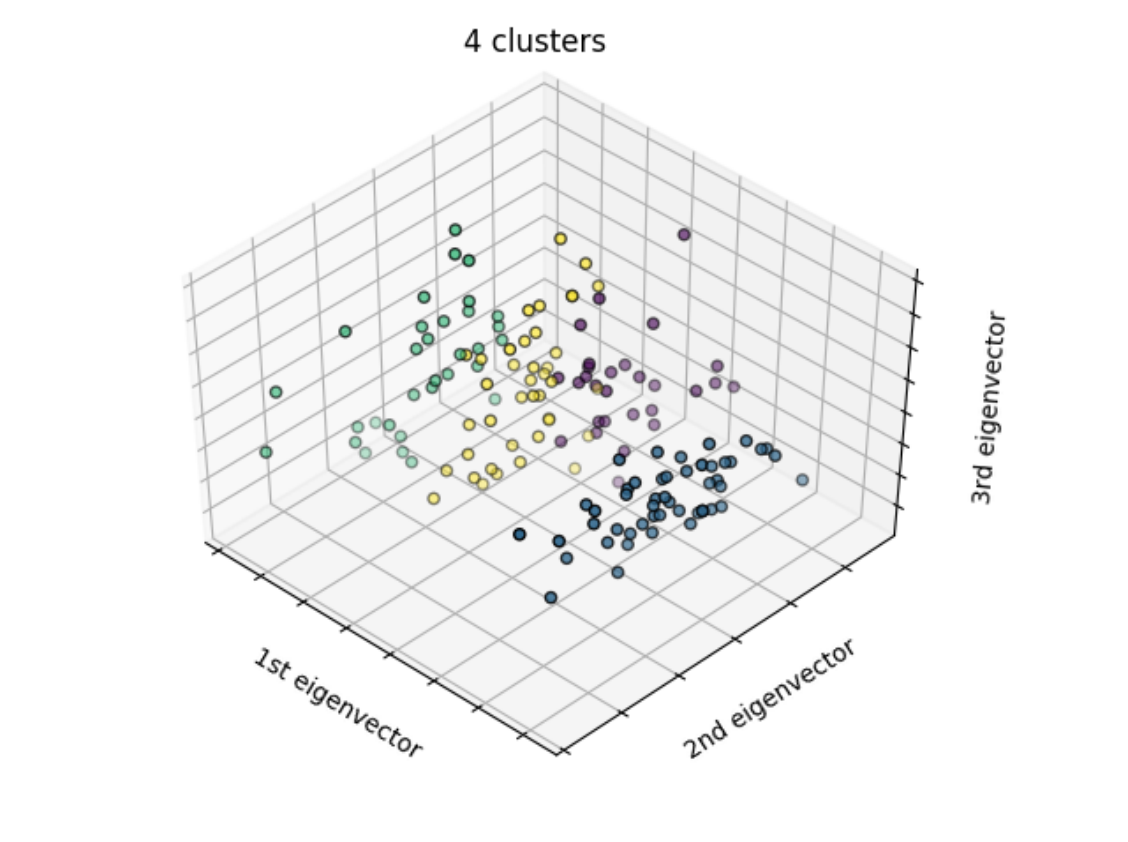
The reduced results are the same as those in Task 2.2. You can refer to Task 2.2 to check the PCA result again. The classification results of k-means on the PCA results are shown as follows:



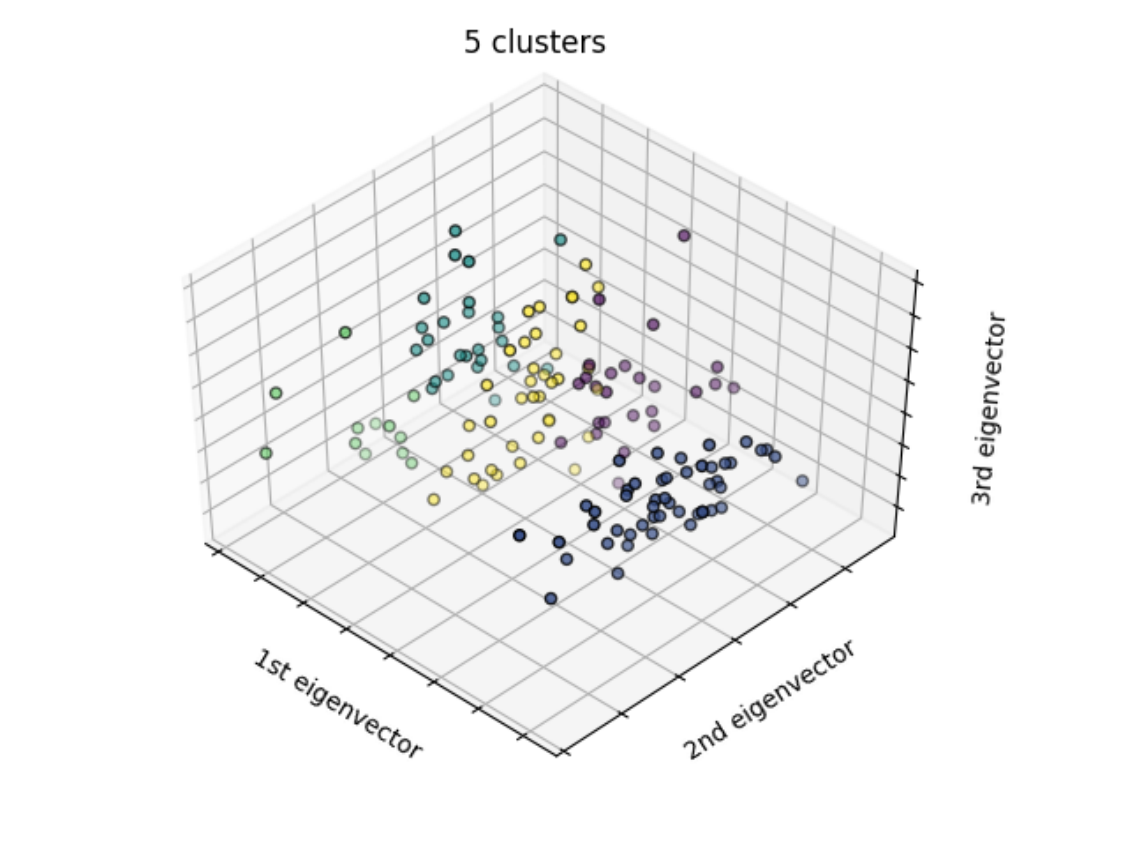


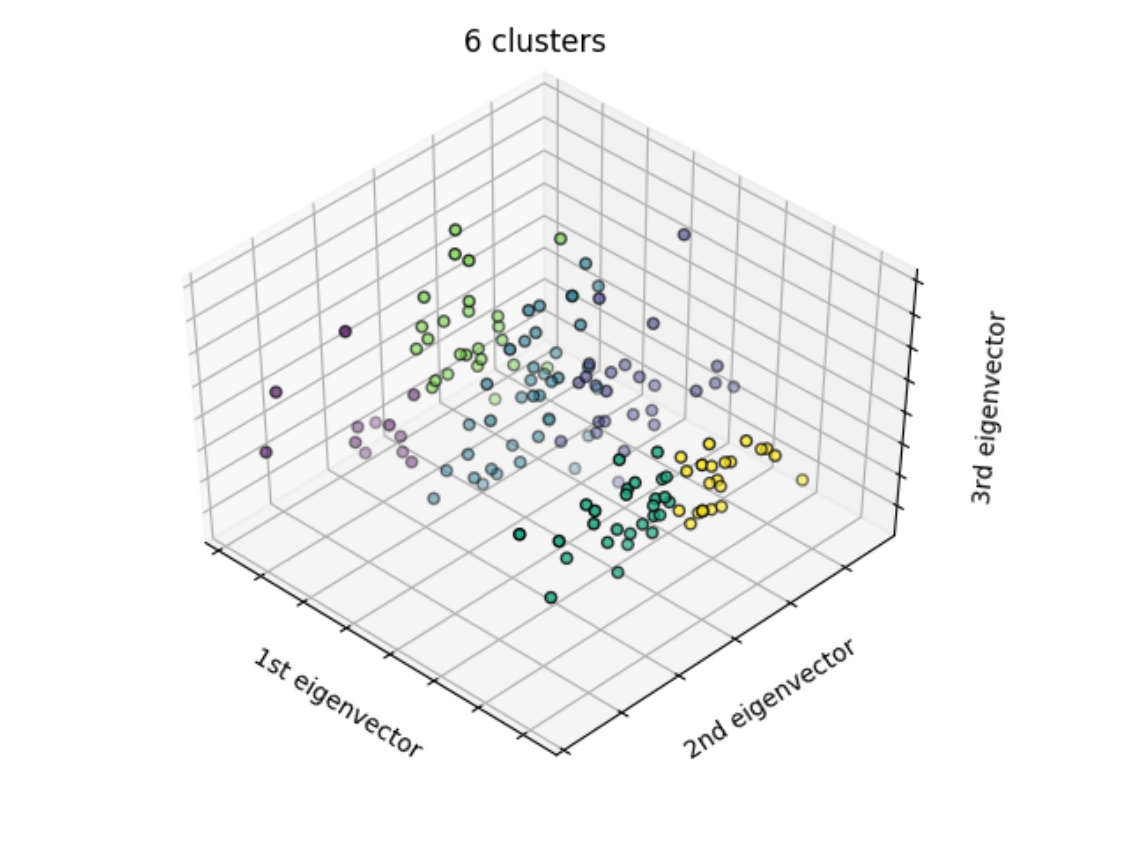
Question 3 continued on next page...





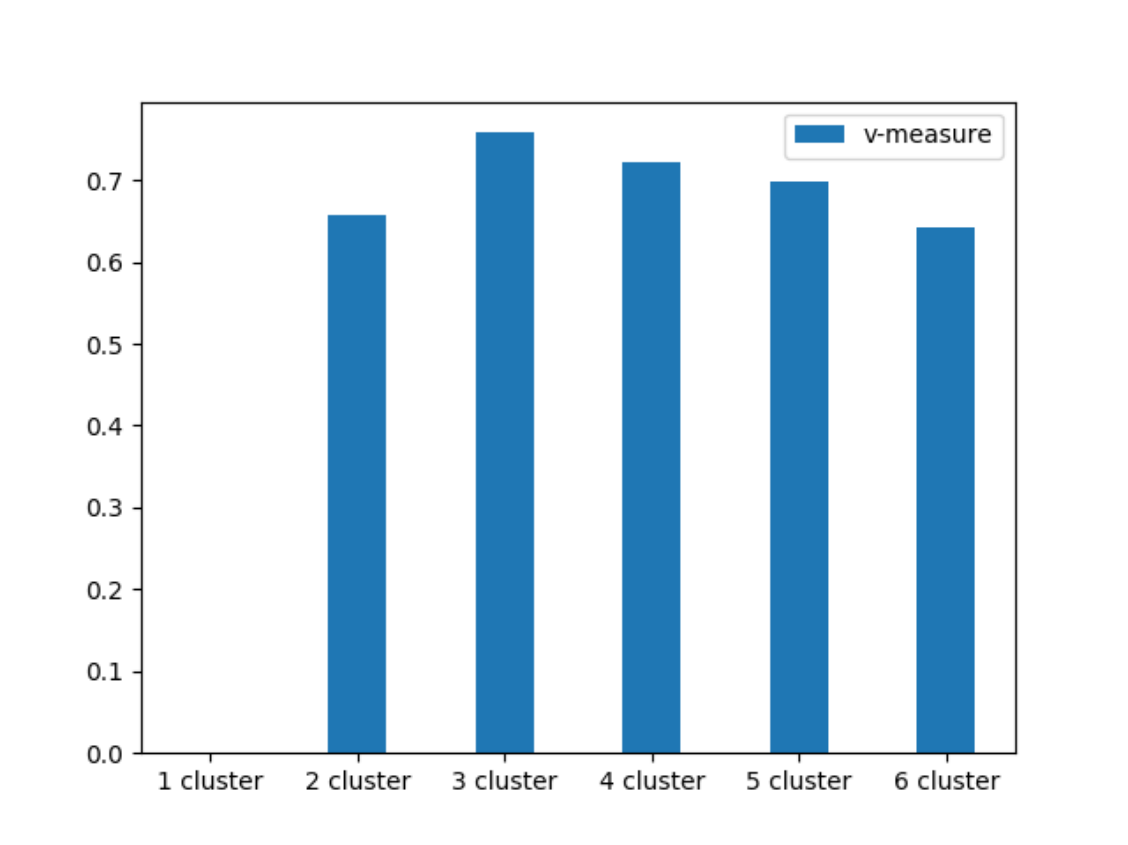
Question 3 continued on next page...





Question 3 continued on next page...

The method to evaluate the model in this question is v-measure, which is the same as Task 3.1. The v-measure results of models with 1-6 clusters are shown in the figures below. According to those figures, it can be concluded that it is 3 clusters that achieves the highest performance. The accuracy of 3 clusters is 0.24.



Question 3 continued on next page...

```
v measure ( k-means 1 clusters ):  -9.095116925863002e-16

v measure ( k-means 2 clusters ):  0.6565191143081123

v measure ( k-means 3 clusters ):  0.7581756800057786

v measure ( k-means 4 clusters ):  0.721920386782096

v measure ( k-means 5 clusters ):  0.696429277767266

v measure ( k-means 6 clusters ):  0.6398842146851936
The accuracy of 3 clusters:  0.24
```

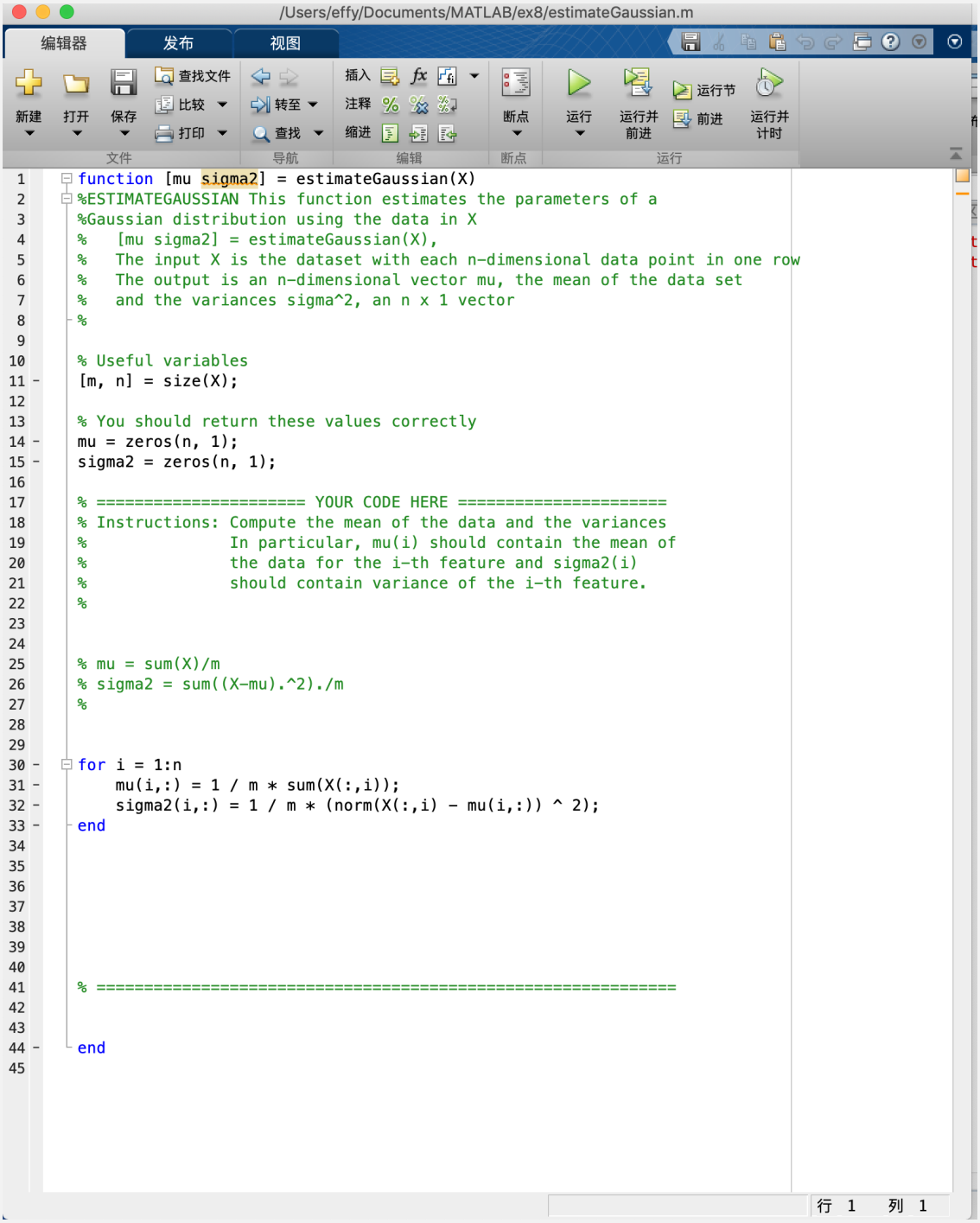
Question 4

Task 4 The Application of Machine Learning

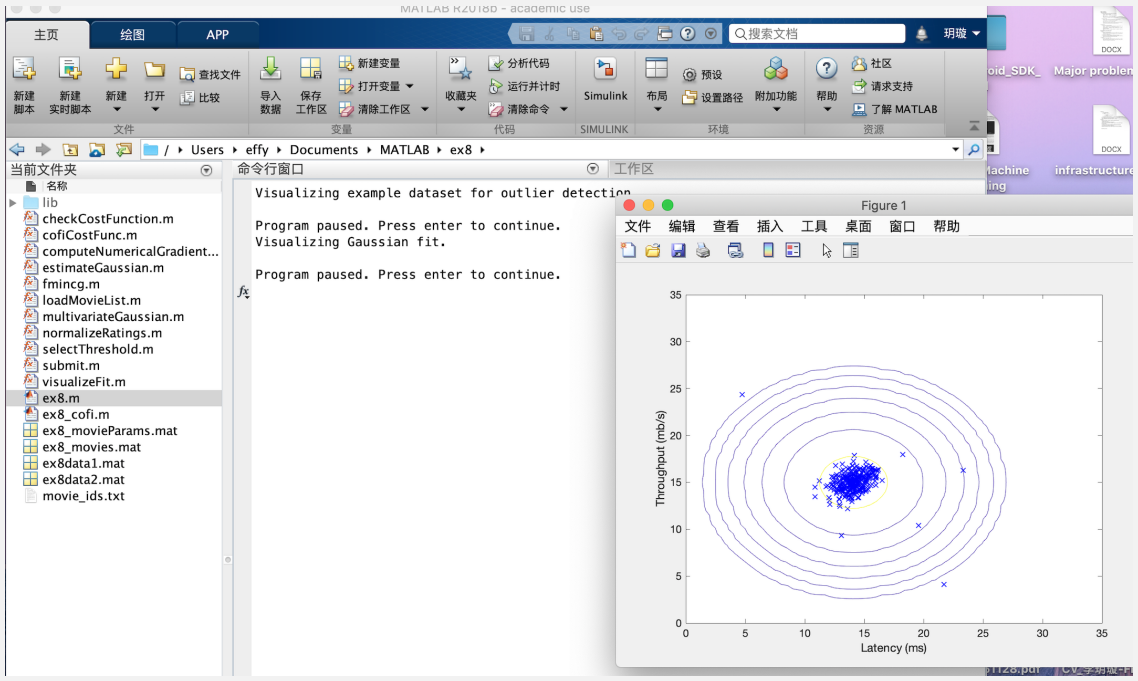
(a) Finish the Coursera Programming Exercises 8 (Anomaly Detection and Recommender Systems) (10')

estimateGuassian.m

Question 4 continued on next page...

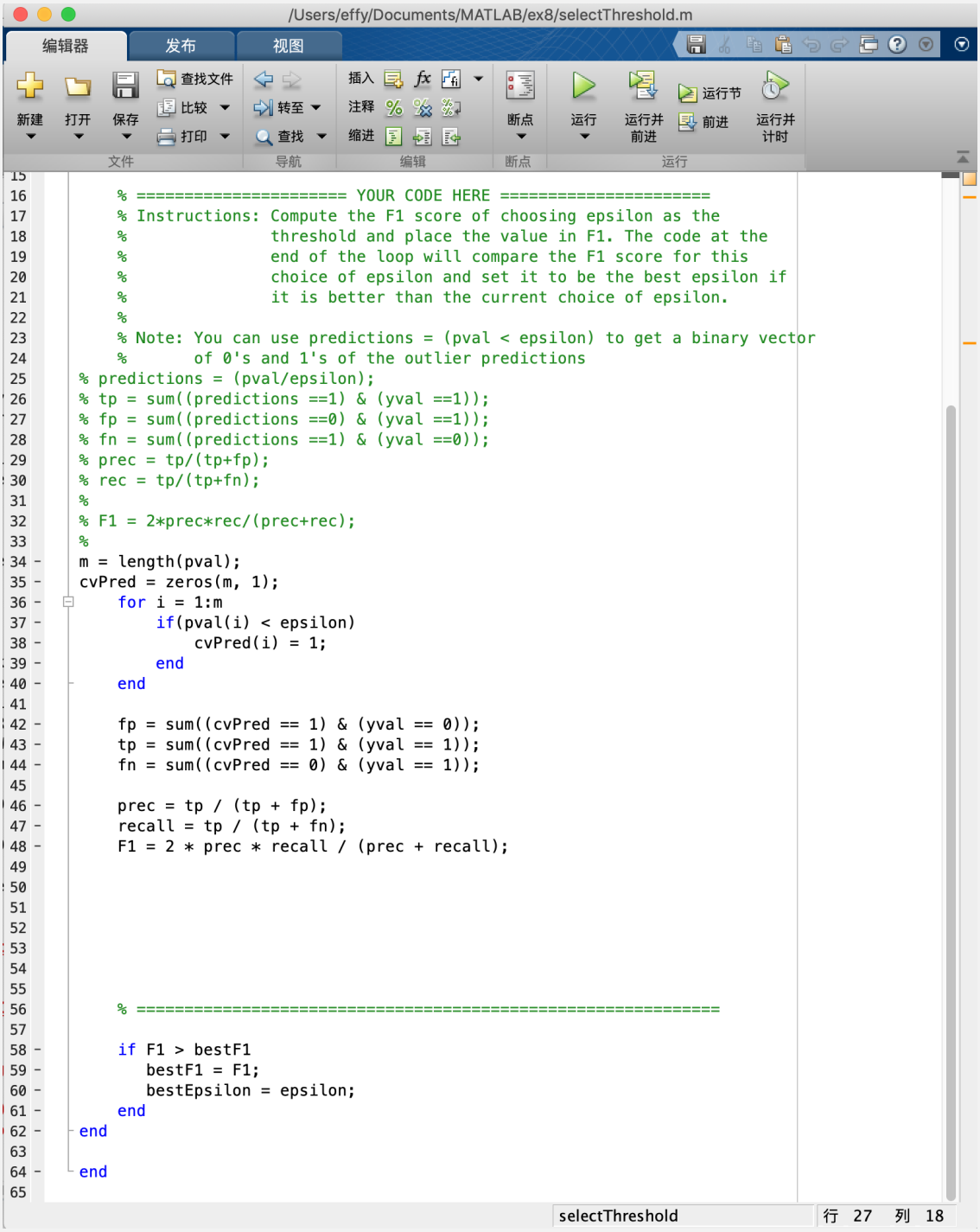


Execution Result

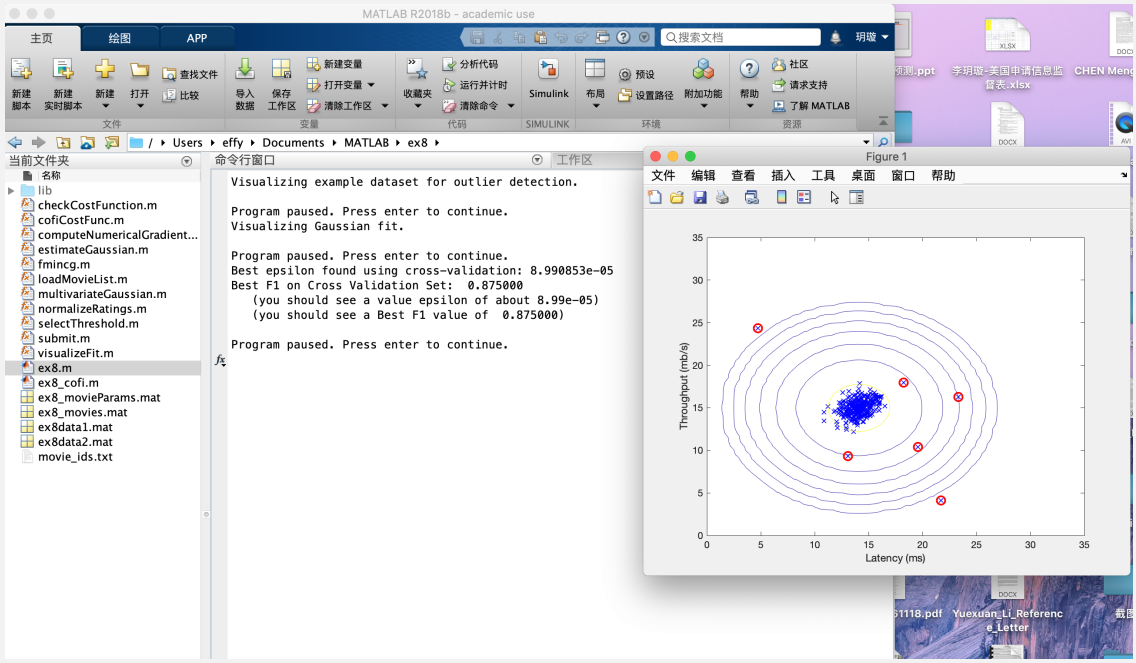


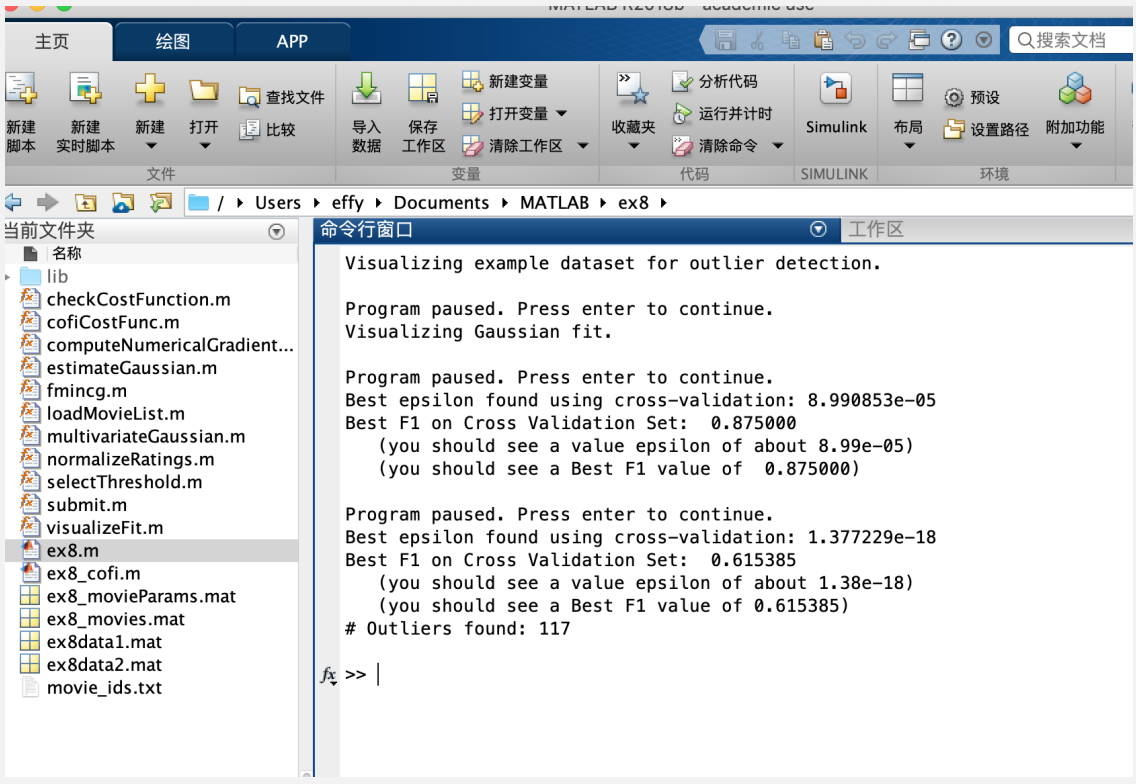
selectThreshold.m

Question 4 continued on next page...

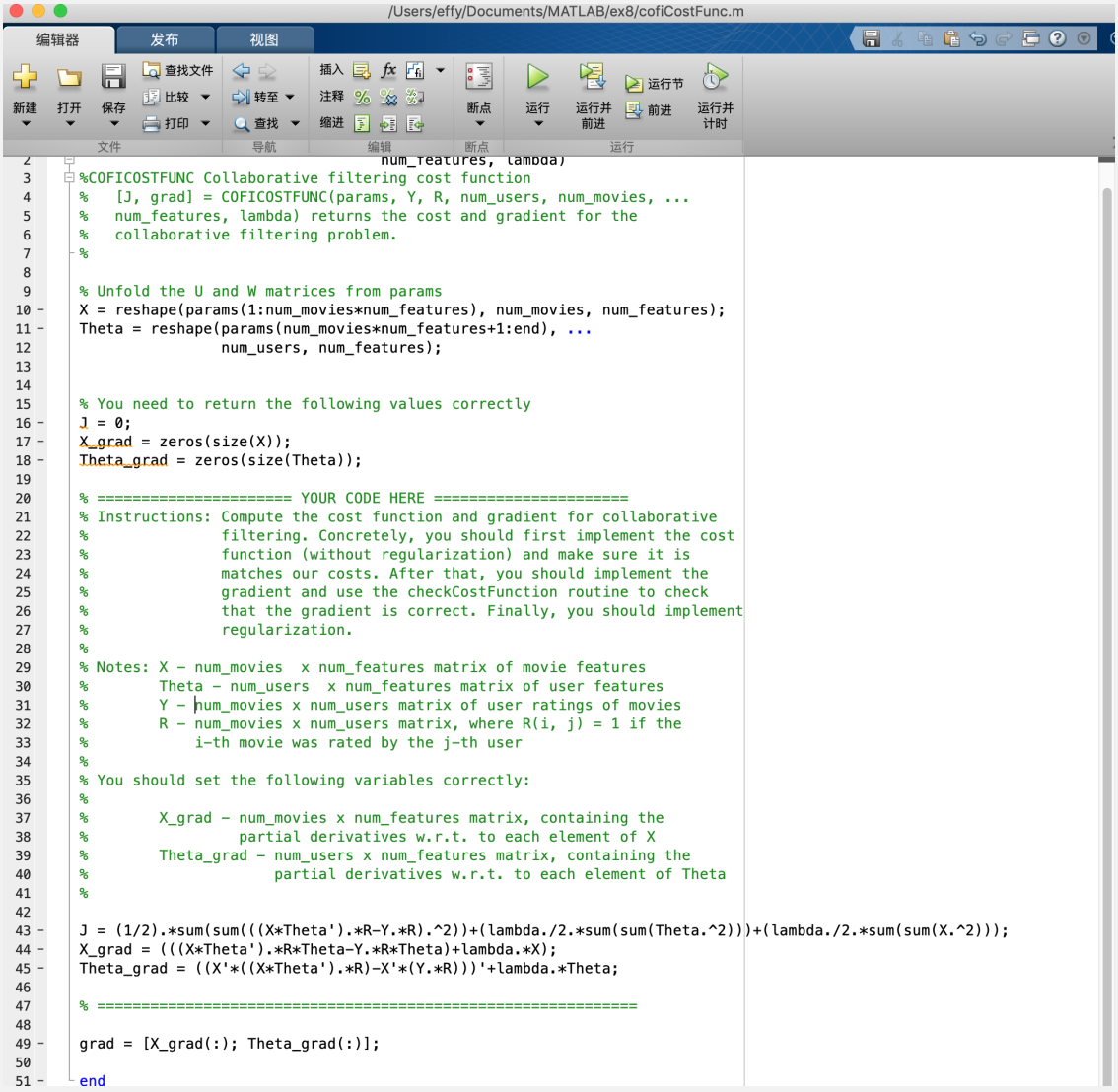


Execution Result





cofiCostFunc.m



Execution Result

```
>> ex8_cofi
Loading movie ratings dataset.

Average rating for movie 1 (Toy Story): 3.878319 / 5


Program paused. Press enter to continue.
Cost at loaded parameters: 22.224604
(this value should be about 22.22)

Program paused. Press enter to continue.

Checking Gradients (without regularization) ...
   -0.4095   -0.4095
    1.0448    1.0448
    1.7044    1.7044
    1.1389    1.1389
   -6.0365   -6.0365
   -1.4956   -1.4956
    1.1993    1.1993
    0.3971    0.3971
  -10.0537  -10.0537
    0.7347    0.7347
    3.9111    3.9111
   -1.3171   -1.3171
    2.0804    2.0804
   -1.9574   -1.9574
   -0.4265   -0.4265
   -2.7934   -2.7934
   -0.2077   -0.2077
   -1.1250   -1.1250
    1.9763    1.9763
   -1.4254   -1.4254
    3.7608    3.7608
   -0.2258   -0.2258
   -1.4078   -1.4078
    4.2922    4.2922
   -1.3301   -1.3301
    8.3475    8.3475
   -0.2618   -0.2618

The above two columns you get should be very similar.
(Left-Your Numerical Gradient, Right-Analytical Gradient)

If your cost function implementation is correct, then
the relative difference will be small (less than 1e-9).
```

Question 4 continued on next page...

```
The above two columns you get should be very similar.
(Left-Your Numerical Gradient, Right-Analytical Gradient)

If your cost function implementation is correct, then
the relative difference will be small (less than 1e-9).

Relative Difference: 1.68957e-12

Program paused. Press enter to continue.
Cost at loaded parameters (lambda = 1.5): 31.344056
(this value should be about 31.34)

Program paused. Press enter to continue.

Checking Gradients (with regularization) ...
    2.4666    2.4666
    6.5979    6.5979
    8.3519    8.3519
    2.0713    2.0713
   -0.6156   -0.6156
    6.9980    6.9980
    2.1364    2.1364
    9.4472    9.4472
   -2.0712   -2.0712
    0.7165    0.7165
   -0.8842   -0.8842
    2.1605    2.1605
    3.5159    3.5159
   -6.7097   -6.7097
   -5.9976   -5.9976
    2.8404    2.8404
   -4.6347   -4.6347
   -0.6331   -0.6331
   -4.0149   -4.0149
  -17.5932  -17.5932
   -0.4677   -0.4677
    0.0785    0.0785
   -1.7391   -1.7391
   -0.0715   -0.0715
   -5.8888   -5.8888
   -0.9253   -0.9253
    1.4390    1.4390

The above two columns you get should be very similar.
(Left-Your Numerical Gradient, Right-Analytical Gradient)

If your cost function implementation is correct, then
```

Question 4 continued on next page...

```
If your cost function implementation is correct, then
the relative difference will be small (less than 1e-9).

Relative Difference: 2.06746e-12

Program paused. Press enter to continue.


New user ratings:
Rated 4 for Toy Story (1995)
Rated 3 for Twelve Monkeys (1995)
Rated 5 for Usual Suspects, The (1995)
Rated 4 for Outbreak (1995)
Rated 5 for Shawshank Redemption, The (1994)
Rated 3 for While You Were Sleeping (1995)
Rated 5 for Forrest Gump (1994)
Rated 2 for Silence of the Lambs, The (1991)
Rated 4 for Alien (1979)
Rated 5 for Die Hard 2 (1990)
Rated 5 for Sphere (1998)

Program paused. Press enter to continue.

Training collaborative filtering...
Iteration     1 | Cost: 3.602961e+05
Iteration     2 | Cost: 1.313692e+05
Iteration     3 | Cost: 1.170872e+05

Iteration     4 | Cost: 8.387548e+04
Iteration     5 | Cost: 6.902281e+04
Iteration     6 | Cost: 5.834475e+04
Iteration     7 | Cost: 4.963457e+04
Iteration     8 | Cost: 4.556481e+04
Iteration     9 | Cost: 4.326882e+04
Iteration    10 | Cost: 4.221671e+04
Iteration    11 | Cost: 4.206001e+04
Iteration    12 | Cost: 4.119046e+04
Iteration    13 | Cost: 4.065083e+04
Iteration    14 | Cost: 4.038739e+04
Iteration    15 | Cost: 4.006824e+04
Iteration    16 | Cost: 3.988804e+04
Iteration    17 | Cost: 3.974650e+04
Iteration    18 | Cost: 3.964423e+04
Iteration    19 | Cost: 3.953689e+04
Iteration    20 | Cost: 3.945622e+04
```

Question 4 continued on next page...

```
Iteration    84 | Cost: 3.895263e+04
Iteration    85 | Cost: 3.895260e+04
Iteration    86 | Cost: 3.895259e+04
Iteration    87 | Cost: 3.895258e+04
Iteration    88 | Cost: 3.895254e+04
Iteration    89 | Cost: 3.895250e+04
Iteration    90 | Cost: 3.895248e+04
Iteration    91 | Cost: 3.895247e+04
Iteration    92 | Cost: 3.895244e+04
Iteration    93 | Cost: 3.895242e+04
Iteration    94 | Cost: 3.895239e+04
Iteration    95 | Cost: 3.895239e+04
Iteration    96 | Cost: 3.895237e+04
Iteration    97 | Cost: 3.895236e+04
Iteration    98 | Cost: 3.895232e+04
Iteration    99 | Cost: 3.895227e+04
Iteration   100 | Cost: 3.895224e+04

Recommender system learning completed.

Program paused. Press enter to continue.

Top recommendations for you:
Predicting rating 5.0 for movie Saint of Fort Washington, The (1993)
Predicting rating 5.0 for movie Entertaining Angels: The Dorothy Day Story (1996)
Predicting rating 5.0 for movie Santa with Muscles (1996)
Predicting rating 5.0 for movie They Made Me a Criminal (1939)
Predicting rating 5.0 for movie Prefontaine (1997)
Predicting rating 5.0 for movie Marlene Dietrich: Shadow and Light (1996)
Predicting rating 5.0 for movie Star Kid (1997)
Predicting rating 5.0 for movie Aiqing wansui (1994)
Predicting rating 5.0 for movie Someone Else's America (1995)
Predicting rating 5.0 for movie Great Day in Harlem, A (1994)


Original ratings provided:
Rated 4 for Toy Story (1995)
Rated 3 for Twelve Monkeys (1995)
Rated 5 for Usual Suspects, The (1995)
Rated 4 for Outbreak (1995)
Rated 5 for Shawshank Redemption, The (1994)
Rated 3 for While You Were Sleeping (1995)
Rated 5 for Forrest Gump (1994)
Rated 2 for Silence of the Lambs, The (1991)
Rated 4 for Alien (1979)
Rated 5 for Die Hard 2 (1990)
Rated 5 for Sphere (1998)
>>
```