Jiayi Llu
9330518335

In [1]:

```
#import path
import os
import numpy as np
root='AReM/'
pathDir =  os.listdir(root)
pathDir.remove('.ipynb_checkpoints')
trainingPath=[]
testPath=[]
for eachDir in pathDir[1:]:
    child = os.path.join('%s%s' % (root, eachDir))
    files= os.listdir(child+'/')
    if eachDir=='bending1' or eachDir=='bending2':
        for i in range(2):
            testPath.append(child+'/dataset'+str(i+1)+'.csv')
        for i in range(2,len(files)):
            trainingPath.append(child+'/dataset'+str(i+1)+'.csv')
    else:
        for i in range(3):
            testPath.append(child+'/dataset'+str(i+1)+'.csv')
        for i in range(3,len(files)):
            trainingPath.append(child+'/dataset'+str(i+1)+'.csv')
```

In [2]:

```
#b)save traindata
import pandas as pd
createVar=locals()
i=0
for csv in trainingPath:
    i=i+1
    data=pd.read_csv(csv,header=None,names=['avg_rss12','var_rss12',
                                            'avg_rss13','var_rss13','avg_rss23',
'var_rss23'],skiprows=5)
    createVar['traindata'+str(i)]=pd.DataFrame(data)

num_train = i;
```

In [3]:

```
#save testdata
i=0
for csv in testPath:
    i=i+1
    createVar['testdata'+str(i)]=pd.read_csv(csv,header=None,names=
                                             ['avg_rss12','var_rss12','avg_rss13
','var_rss13','avg_rss23','var_rss23'],
                                             skiprows=5)
num_test = i;
```

In [595]:

```
print("c)i Some popular time-domain features: Mean, Median, Standard Deviation, Variance, Root Mean Square, Averaged derivatives, minimum, maximum etc. ")
```

```
c)i Some popular time-domain features: Mean, Median, Standard Deviat
ion, Variance, Root Mean Square, Averaged derivatives, minimum, maxi
mum etc.
```

In [5]:

```
#c) ii def a function that give a row of 88*42
def give7features(td):
    td_mean = td.mean()
    td_mean_array = np.zeros(shape=(1,6))
    for i in range(6):
        td_mean_array[0,i]=td_mean.iloc[i]

    td_max = td.max()
    td_max_array = np.zeros(shape=(1,6))
    for i in range(6):
        td_max_array[0,i]=td_max.iloc[i]

    td_min = td.min()
    td_min_array = np.zeros(shape=(1,6))
    for i in range(6):
        td_min_array[0,i]=td_min.iloc[i]

    td_median = td.median()
    td_median_array = np.zeros(shape=(1,6))
    for i in range(6):
        td_median_array[0,i]=td_median.iloc[i]

    td_std = td.std()
    td_std_array = np.zeros(shape=(1,6))
    for i in range(6):
        td_std_array[0,i]=td_std.iloc[i]

    td_1stquartile = td.quantile(q=0.25)
    td_1stquartile_array = np.zeros(shape=(1,6))
    for i in range(6):
        td_1stquartile_array[0,i]=td_1stquartile.iloc[i]

    td_3stquartile = td.quantile(q=0.75)
    td_3stquartile_array = np.zeros(shape=(1,6))
    for i in range(6):
        td_3stquartile_array[0,i]=td_3stquartile.iloc[i]

    td_7=np.concatenate((td_min_array,td_max_array,td_mean_array,td_median_array
,
                         td_std_array,td_1stquartile_array,td_3stquartile_array)
,axis=0)

    #td_7=np.concatenate((td_min_array,td_max_array,td_mean_array,td_median_arra
y,td_std_array,td_1stquartile_array,td_3stquartile_array),axis=1)
    return td_7.reshape([1,42],order='F')
```

In [6]:

```
#stack
dataset=np.empty(shape=[0, 42])
for i in range(num_train):
    i=i+1
    dataset = np.concatenate((dataset,give7features(createVar['traindata'+str(i)
])),axis=0)
for j in range(num_test):
    j=j+1
    dataset = np.concatenate((dataset,give7features(createVar['testdata'+str(j)]
)),axis=0)
```

In [8]:

```
#1(c)iii extracted from time series 1, 2, and 6
index_1=[0,1,2,7,8,9,35,36,37]
extracted_1 = dataset[:,index_1]
```

In [9]:

```
temp_class_1 = np.zeros(shape=(1,88))
```

In [10]:

```
#d) i
for i in range(0,9):
    temp_class_1[0,i]=1

#for i in range(69,73):
#   temp_class_1[0,i]=1
temp_class_1 = temp_class_1.T
```

In [11]:

```
new_extrated_class=np.column_stack((extracted_1,temp_class_1))
```

In [12]:

```
new_extrated_trclass = new_extrated_class[0:69]
```

In [13]:

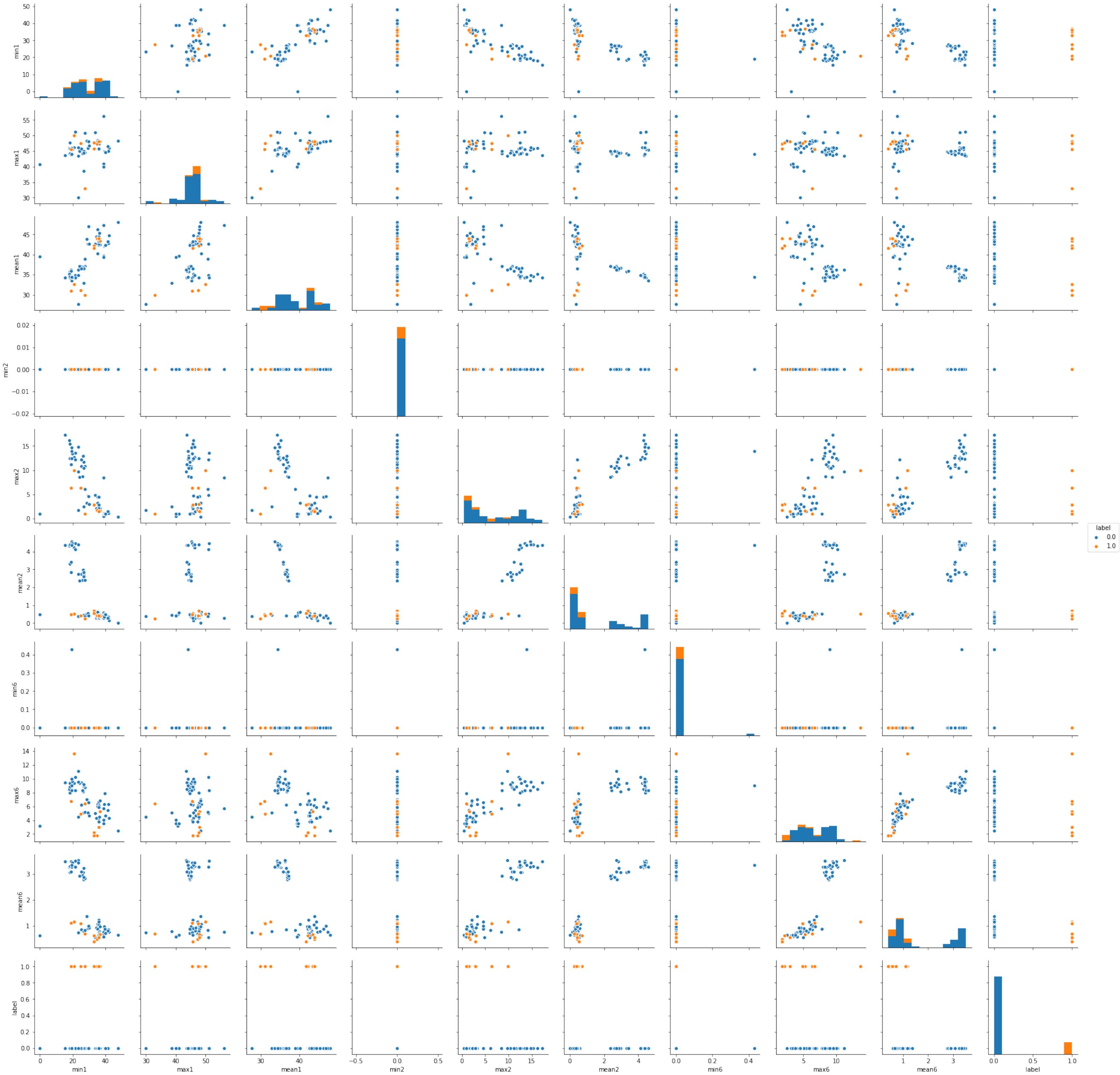
```
df_binary_1 = pd.DataFrame(new_extrated_trclass,columns = ['min1','max1','mean1'
,'min2','max2','mean2',
                                                             'min6','max6','mean6'
,'label'])
```

In [14]:

```
import seaborn as sns
%matplotlib inline
sns.pairplot(df_binary_1, hue="label")
```

Out[14]:

<seaborn.axisgrid.PairGrid at 0x1100e6390>



In [15]:

```
#b) ii split data into n pieces
def splitdata(td,n):
    num_row = td.iloc[:,0].size
    for c in range (n):
        c=c+1
        if(c!=n):
            createVar['traindata_s'+str(c)] = td[int(num_row/n)*(c-1):int(num_ro
w/n)*(c)]
        else:
            createVar['traindata_s'+str(c)] = td[int(num_row/n)*(c-1):]

    tup=(createVar['traindata_s'+str(1)],)
    for c in range (1,n):
        c=c+1
        tup=tup+(createVar['traindata_s'+str(c)],)
    return tup
```

In [16]:

```
for i in range(num_train):
    i=i+1
    createVar['traindata'+str(i)+'1'],createVar['traindata'+str(i)+'2'] = splitd
ata(createVar['traindata'+str(i)],2)
```

In [17]:

```
#stack2
dataset_2=np.empty(shape=[0, 42])
for i in range(num_train):
    i=i+1
    dataset_2 = np.concatenate((dataset_2,give7features(createVar['traindata'+st
r(i)+'1'])),axis=0)
    dataset_2 = np.concatenate((dataset_2,give7features(createVar['traindata'+st
r(i)+'2'])),axis=0)
```

In [18]:

```
index_2=[0,1,2,7,8,9,35,36,37]
extracted_2 = dataset_2[:,index_2]
```

In [19]:

```
temp_class_2 = np.zeros(shape=(1,num_train*2))
```

In [20]:

```
for i in range(0,18):
    temp_class_2[0,i]=1

#for i in range(69,73):
#   temp_class_1[0,i]=1
temp_class_2 = temp_class_2.T
```

In [21]:

```
new_extrated_class_2=np.column_stack((extracted_2,temp_class_2))
```

In [22]:

```
df_binary_2 = pd.DataFrame(new_extrated_class_2,columns = ['min1','max1','mean1'
,'min2','max2','mean2',
                                                            'min6','max6','mean6'
,'label'])
```

In [23]:

```
df_binary_2.shape
```

Out[23]:

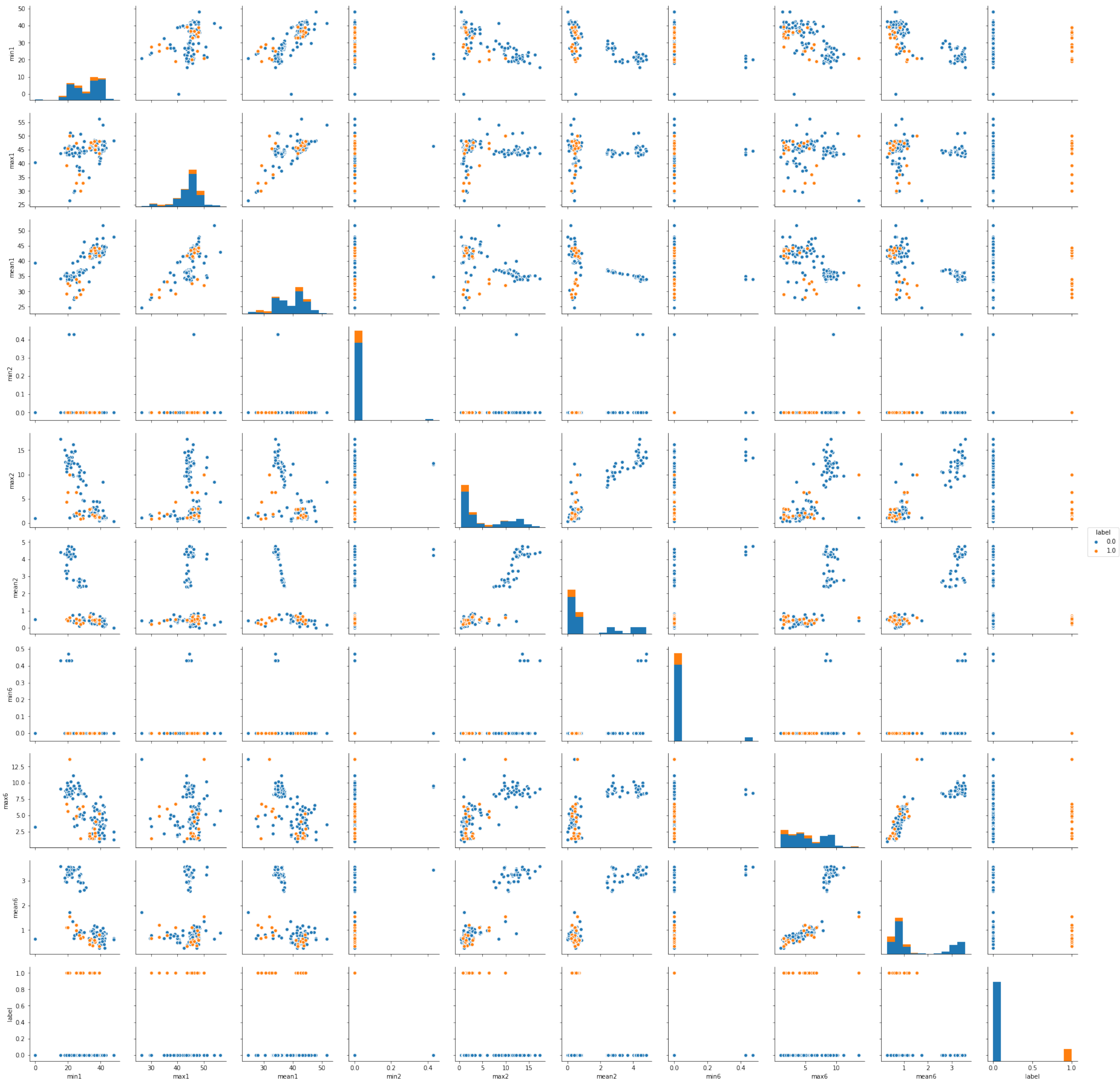
```
(138, 10)
```

In [24]:

```
sns.pairplot(df_binary_2, hue="label")
```

Out[24]:

```
<seaborn.axisgrid.PairGrid at 0x1a202a9320>
```



In [25]:

```
print("Compared to d) i, the number of data increase but so as the noise")
```

Compared to d) i, the number of data increase but so as the noise

In [26]:

```
#d) iii every traindata need a empty array to store 42-feartures, n pieces n*42
for i in range(20):
    i=i+1
    createVar['dataset_'+str(i)]=np.empty(shape=[0, 42])
```

In [27]:

```
#split every train data(from traindata1 to traindata69) into n pieces, save into
p1 to p69, each pi have n dataframs. p are tuples
def finaldata(n):
    for i in range(num_train):
        i=i+1
        createVar['p'+str(i)]= splitdata(createVar['traindata'+str(i)],n)


    for i in range(num_train):
        i=i+1
        for j in range(n):
            j=j+1
            createVar['dataset_'+str(n)] = np.concatenate((createVar['dataset_'+
str(n)],
                                                          give7features(createV
ar['p'+str(i)][j-1])),axis=0)
```

In [28]:

```
for i in range(20):
    i=i+1
    finaldata(i)
```

In [29]:

```
dataset_20.shape
```

Out[29]:

```
(1380, 42)
```

In [30]:

```
#extract features
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression
def featureExtract(x,y,f):
    index=[]
    model = LogisticRegression( )
    rfe = RFE(model,f)

    rfe = rfe.fit(x,y)
    x=rfe.support_
    for i in range(f):
        index.append(np.where(x==True)[0][i])
    return index
```

In [31]:

```
for i in range(20):
    i=i+1
    createVar['y_'+str(i)]=  np.zeros(shape=(1,69*i))
for i in range(20):
    i=i+1
    for j in range(0,9*i):
        createVar['y_'+str(i)][0,j]=1
for i in range(20):
    i=i+1
    createVar['y_'+str(i)] = createVar['y_'+str(i)].T
```

In [32]:

```
print("fit all features")
from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score
acc_42 = list()
for i in range(20):
    i=i+1
    createVar['clf_'+str(i)] = LogisticRegression(random_state = 0)
    createVar['clf_'+str(i)].fit(createVar['dataset_'+str(i)],createVar['y_'+str
(i)].ravel())
    createVar['y_pred_'+str(i)]= createVar['clf_'+str(i)].predict(createVar['dat
aset_'+str(i)])
    createVar['test_acc_'+str(i)]=accuracy_score(createVar['y_'+str(i)],createVa
r['y_pred_'+str(i)])
    acc_42.append(createVar['test_acc_'+str(i)])
```

```
fit all features
```

In [597]:

```
for i in range (20):
    i=i+1
    print("the predicted class for training set when l = "+str(i)+" is:")
    print( createVar['y_pred_'+str(i)])
    print( "train accuracy when l = "+str(i)+"is:")
    print( createVar['test_acc_'+str(i)])
```

```
the predicted class for training set when l = 1 is:
[1. 0. 1. 1. 0. 1. 0. 1. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
train accuracy when l = 1is:
0.9420289855072463
the predicted class for training set when l = 2 is:
[0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 1. 1. 0. 0. 1. 1. 1. 0. 0. 0. 0. 0. 0
```

```
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
train accuracy when l = 2is:
0.9130434782608695
the predicted class for training set when l = 3 is:
[1. 1. 0. 1. 0. 0. 1. 1. 0. 1. 0. 1. 0. 0. 0. 1. 0. 1. 0. 1. 1. 1. 1
. 1.
 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
train accuracy when l = 3is:
0.927536231884058
the predicted class for training set when l = 4 is:
[0. 0. 1. 1. 1. 1. 1. 0. 1. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0
. 1.
 0. 1. 0. 1. 1. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
```

```
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
train accuracy when l = 4is:
0.9130434782608695
the predicted class for training set when l = 5 is:
[1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 1. 1. 1. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0
. 0.
 0. 1. 1. 0. 1. 0. 0. 1. 0. 1. 1. 1. 0. 1. 1. 1. 0. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0.]
train accuracy when l = 5is:
0.9246376811594202
the predicted class for training set when l = 6 is:
[1. 0. 1. 1. 1. 1. 1. 1. 0. 1. 1. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 1. 1. 0. 0. 1. 0. 0. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1
. 0.
 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
```

```
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0.]
train accuracy when l = 6is:
0.9154589371980676
the predicted class for training set when l = 7 is:
[1. 1. 1. 0. 1. 1. 1. 1. 0. 1. 0. 1. 1. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 0. 1. 1. 0. 1. 1. 0. 1
. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
. 0.
 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0
. 0.
 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
```

```
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0.]
train accuracy when l = 7is:
0.917184265010352
the predicted class for training set when l = 8 is:
[1. 0. 1. 1. 0. 1. 1. 1. 0. 1. 1. 0. 1. 0. 1. 0. 1. 0. 1. 1. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 1. 1. 0. 0. 0. 0
. 0.
 0. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 0. 0. 0. 0. 1
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 1. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
```

```
. 0.
 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.]
train accuracy when l = 8is:
0.9112318840579711
the predicted class for training set when l = 9 is:
[1. 0. 0. 1. 0. 1. 0. 1. 1. 0. 1. 1. 1. 0. 1. 0. 1. 0. 1. 0. 1. 1. 1
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 1. 1
. 1.
 0. 1. 0. 0. 0. 0. 0. 0. 1. 1. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1
. 1.
 0. 1. 1. 0. 0. 0. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
```

```
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
train accuracy when l = 9is:
0.9114331723027376
the predicted class for training set when l = 10 is:
[1. 0. 1. 1. 1. 1. 1. 1. 0. 1. 0. 1. 1. 1. 1. 1. 1. 1. 0. 0. 1. 0. 1
. 0.
 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 1. 1. 1. 1. 0. 0. 0. 1. 0. 0. 0. 0. 1. 1. 0. 0. 1. 1. 1. 1. 1
. 1.
 0. 0. 1. 1. 0. 0. 1. 1. 0. 0. 1. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
```

```
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
train accuracy when l = 10is:
0.9130434782608695
the predicted class for training set when l = 11 is:
[1. 0. 1. 1. 1. 0. 1. 1. 1. 0. 1. 0. 1. 1. 1. 0. 1. 1. 1. 0. 0. 0. 1
. 0.
 1. 0. 1. 1. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 1. 0. 1. 0. 0. 0. 1. 0. 0. 1. 1. 1. 0
. 0.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 1. 1. 1. 0. 1. 1. 0. 0. 0
. 0.
 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
```

```
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
train accuracy when l = 11is:
0.919631093544137
the predicted class for training set when l = 12 is:
[1. 0. 1. 1. 1. 0. 0. 1. 0. 1. 1. 1. 0. 1. 0. 1. 1. 0. 1. 1. 0. 1. 0
. 0.
 1. 0. 0. 1. 1. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 1. 1. 1. 0. 0. 1. 0. 0
. 1.
 0. 0. 1. 0. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 0. 1. 1. 0
. 1.
 1. 0. 1. 1. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
```

```
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 1. 1. 0. 0. 0. 1. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
train accuracy when l = 12is:
0.9057971014492754
the predicted class for training set when l = 13 is:
[1. 0. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 0. 1. 1. 1. 1. 0. 1. 1. 1. 0
. 1.
 0. 1. 1. 0. 0. 1. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
```

```
. 0.
 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 1. 1. 0
. 1.
 0. 0. 1. 0. 0. 1. 0. 0. 1. 1. 1. 0. 0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 0
. 0.
 1. 1. 1. 1. 0. 1. 0. 1. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 1. 0. 0
. 0.
 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
```

```
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0.]
train accuracy when l = 13is:
0.9175027870680045
the predicted class for training set when l = 14 is:
[1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 0. 1. 1. 1. 1. 1. 1
. 1.
 1. 1. 0. 0. 1. 0. 0. 1. 0. 1. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1
. 1.
 1. 1. 1. 0. 1. 0. 0. 0. 1. 1. 0. 1. 0. 0. 1. 1. 1. 0. 0. 0. 1. 1. 1
. 0.
 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 0. 1. 1. 0. 0. 1. 0. 0. 0
. 0.
 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
```

```
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 1. 1. 1. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0.]
train accuracy when l = 14is:
0.9130434782608695
the predicted class for training set when l = 15 is:
```

```
[1. 0. 1. ... 0. 0. 0.]
train accuracy when l = 15is:
0.9130434782608695
the predicted class for training set when l = 16 is:
[1. 0. 0. ... 0. 0. 0.]
train accuracy when l = 16is:
0.9021739130434783
the predicted class for training set when l = 17 is:
[1. 0. 0. ... 0. 0. 0.]
train accuracy when l = 17is:
0.9053708439897699
the predicted class for training set when l = 18 is:
[1. 0. 0. ... 0. 0. 0.]
train accuracy when l = 18is:
0.9017713365539453
the predicted class for training set when l = 19 is:
[1. 0. 0. ... 0. 0. 0.]
train accuracy when l = 19is:
0.9099923722349351
the predicted class for training set when l = 20 is:
[1. 0. 0. ... 0. 0. 0.]
train accuracy when l = 20is:
0.913768115942029
```

In [34]:

```
print("the accuracy is max when l = "+str(acc_42.index(max(acc_42))+1))
print("which is " +str(max(acc_42)))
```

```
the accuracy is max when l = 1
which is 1.0
```

In [35]:

```
import rpy2.robjects as ro
from rpy2.robjects import pandas2ri
f=ro.r['glm']
```

In [36]:

```
for i in range(20):
    i=i+1
    createVar['new_dataset_'+str(i)]=np.column_stack((createVar['dataset_'+str(i
)],createVar['y_'+str(i)]))
    createVar['df_bi_'+str(i)] = pd.DataFrame(createVar['new_dataset_'+str(i)],c
olumns = ['min1','max1',

'mean1','median1','std1',

'one_quartile1','th_quartile1',
                                                       'min2','max2','mean2','medi
an2','std2','one_quartile2','th_quartile2',
                                                       'min3','max3','mean3','medi
an3','std3','one_quartile3','th_quartile3',
                                                       'min4','max4','mean4','medi
an4','std4','one_quartile4','th_quartile4',
                                                       'min5','max5','mean5','medi
an5','std5','one_quartile5','th_quartile5',
                                                       'min6','max6','mean6','medi
an6','std6','one_quartile6','th_quartile6',
                                                       'label'])
```

In [37]:

```
df_bi_1.shape
```

Out[37]:

```
(69, 43)
```

In [38]:

```
pandas2ri.activate()
for i in range(20):
    i=i+1
    feature=createVar['df_bi_'+str(i)].columns[0:42]
    mylogit = f(formula="label~(min1+max1+mean1+median1+std1+one_quartile1+th_qu
artile1+min2+max2+mean2+median2+std2+one_quartile2+th_quartile2+min3+max3+mean3+
median3+std3+one_quartile3+th_quartile3+min4+max4+mean4+median4+std4+one_quartil
e4+th_quartile4+min5+max5+mean5+median5+std5+one_quartile5+th_quartile5+min6+max
6+mean6+median6+std6+one_quartile6+th_quartile6)",data=createVar['df_bi_'+str(i)
],family=ro.r('binomial(link="logit")'))
    print("p values when for all features when l = " +str(i))
    print(ro.r.summary(mylogit)[-6])
```

```
p values when for all features when l = 1
                  Estimate Std. Error       z value  Pr(>|z|)
(Intercept)    -55.3749022 1722887.96 -3.214074e-05 0.9999744
min1            -0.1770208   23016.98 -7.690878e-06 0.9999939
max1            -6.7758958   97554.17 -6.945777e-05 0.9999446
mean1           40.7412300 1069233.40  3.810321e-05 0.9999696
```

```
median1          1.7412045  232545.55  7.487585e-06 0.9999940
std1            14.4090250  434499.08  3.316238e-05 0.9999735
one_quartile1  -14.3532758  406773.96 -3.528563e-05 0.9999718
th_quartile1   -23.7116311  544872.45 -4.351776e-05 0.9999653
max2            -8.6067826  125244.28 -6.871997e-05 0.9999452
mean2           10.9098357 2624698.26  4.156606e-06 0.9999967
median2         35.6546290  967273.90  3.686094e-05 0.9999706
std2           137.8060698 2342352.03  5.883235e-05 0.9999531
one_quartile2   46.9130737 1004843.26  4.668696e-05 0.9999627
th_quartile2   -77.9110596 1195469.41 -6.517194e-05 0.9999480
min3            -7.6102654   52498.62 -1.449613e-04 0.9998843
max3            -1.1958961   83694.94 -1.428875e-05 0.9999886
mean3           14.3792237  739885.26  1.943440e-05 0.9999845
median3         32.5222711  258507.59  1.258078e-04 0.9998996
std3           -20.7636380 1005973.36 -2.064035e-05 0.9999835
one_quartile3  -35.1473335  422555.21 -8.317808e-05 0.9999336
th_quartile3   -12.4252665  430044.39 -2.889299e-05 0.9999769
max4            16.3922563  126039.45  1.300566e-04 0.9998962
mean4          -73.7168854 2577899.06 -2.859572e-05 0.9999772
median4        -84.7028816 2585280.66 -3.276351e-05 0.9999739
std4            17.4110613 2033433.99  8.562393e-06 0.9999932
one_quartile4   70.0897147  973654.75  7.198621e-05 0.9999426
th_quartile4   -44.4164349 1274446.95 -3.485154e-05 0.9999722
min5            -5.9160312   78212.43 -7.564055e-05 0.9999396
max5            11.8466963   79491.35  1.490313e-04 0.9998811
mean5          -59.7823369 1408414.17 -4.244656e-05 0.9999661
median5         17.2175198  293794.67  5.860392e-05 0.9999532
std5           -36.1585384  483756.04 -7.474540e-05 0.9999404
one_quartile5    5.9948190  557581.49  1.075147e-05 0.9999914
th_quartile5    36.4839887  521836.35  6.991462e-05 0.9999442
min6            18.0679010 1339328.56  1.349027e-05 0.9999892
max6             5.4455133  170732.02  3.189509e-05 0.9999746
mean6          426.3700413 2571370.20  1.658143e-04 0.9998677
median6        -38.1667300 2598463.04 -1.468819e-05 0.9999883
std6          -390.6254724 1970851.22 -1.982014e-04 0.9998419
one_quartile6 -250.0942302 1126295.86 -2.220502e-04 0.9998228
th_quartile6    39.1767540 1556602.07  2.516812e-05 0.9999799

p values when for all features when l = 2
                Estimate Std. Error       z value  Pr(>|z|)
(Intercept)    26.846777  910336.83  2.949104e-05 0.9999765
min1           -6.110295   13321.85 -4.586672e-04 0.9996340
max1            7.415199  131445.08  5.641291e-05 0.9999550
mean1         -33.690139  333034.44 -1.011611e-04 0.9999193
median1        -7.216812  152462.99 -4.733484e-05 0.9999622
std1          -14.617440  266403.68 -5.486951e-05 0.9999562
one_quartile1  19.379003  211320.83  9.170418e-05 0.9999268
th_quartile1   12.599653   90357.60  1.394421e-04 0.9998887
min2         -235.143243 2109091.59 -1.114903e-04 0.9999110
max2           41.681148  204275.25  2.040440e-04 0.9998372
mean2        -341.418002 1911894.13 -1.785758e-04 0.9998575
median2       112.366444  689823.94  1.628915e-04 0.9998700
std2         -587.867915 2032846.50 -2.891846e-04 0.9997693
```

```
one_quartile2  -25.541117  617691.63 -4.134930e-05 0.9999670
th_quartile2   366.946372  875370.91  4.191896e-04 0.9996655
min3            -7.289138   56526.32 -1.289512e-04 0.9998971
max3            23.563071   83600.99  2.818516e-04 0.9997751
mean3         -105.062579  761691.26 -1.379333e-04 0.9998899
median3         18.465686  311419.93  5.929513e-05 0.9999527
std3           -69.447415  407485.26 -1.704293e-04 0.9998640
one_quartile3   22.411327  298350.04  7.511756e-05 0.9999401
th_quartile3    40.098593  171173.00  2.342577e-04 0.9998131
min4           472.367238 1324124.40  3.567393e-04 0.9997154
max4            -2.359768  103214.05 -2.286286e-05 0.9999818
mean4         -300.334230 1663881.32 -1.805022e-04 0.9998560
median4       -185.382749  964997.21 -1.921070e-04 0.9998467
std4            88.350440 1113176.73  7.936785e-05 0.9999367
one_quartile4  172.640582  338226.54  5.104288e-04 0.9995927
th_quartile4   115.675588  438149.74  2.640093e-04 0.9997894
min5           -14.001666   45445.58 -3.080974e-04 0.9997542
max5            35.274906  104045.52  3.390334e-04 0.9997295
mean5            5.571041  547296.07  1.017921e-05 0.9999919
median5        -15.665552  131198.43 -1.194035e-04 0.9999047
std5          -111.572321  327694.05 -3.404771e-04 0.9997283
one_quartile5  -29.875917  198612.08 -1.504235e-04 0.9998800
th_quartile5    32.091090  291139.33  1.102259e-04 0.9999121
min6          -261.978519 2512067.16 -1.042880e-04 0.9999168
max6           -14.431041  350037.57 -4.122712e-05 0.9999671
mean6         -184.694292 1658948.13 -1.113322e-04 0.9999112
median6         90.031582  880805.80  1.022150e-04 0.9999184
std6             3.977006 2354823.52  1.688877e-06 0.9999987
one_quartile6   46.143975  524567.17  8.796581e-05 0.9999298
th_quartile6   106.015729  846227.29  1.252804e-04 0.9999000
```

p values when for all features when l = 3

```
                 Estimate Std. Error       z value  Pr(>|z|)
(Intercept)    1294.59724 287339.518  4.505462e-03 0.9964052
min1            -21.66935   6700.080 -3.234193e-03 0.9974195
max1             30.29387  10091.583  3.001895e-03 0.9976048
mean1          -165.59296  95012.439 -1.742856e-03 0.9986094
median1          60.85039  30995.180  1.963221e-03 0.9984336
std1           -182.86851  49631.624 -3.684516e-03 0.9970602
one_quartile1   -24.23158  40251.347 -6.020066e-04 0.9995197
th_quartile1     50.95557  28357.317  1.796911e-03 0.9985663
min2             87.60733 613193.696  1.428706e-04 0.9998860
max2            244.48370  26931.458  9.077997e-03 0.9927569
mean2           401.02554 283137.914  1.416361e-03 0.9988699
median2        -195.45039 118923.769 -1.643493e-03 0.9986887
std2          -3204.84039 301484.455 -1.063020e-02 0.9915185
one_quartile2  -325.52528 104597.783 -3.112162e-03 0.9975169
th_quartile2    588.11275 118328.267  4.970180e-03 0.9960344
min3            -40.11154  11327.613 -3.541041e-03 0.9971747
max3             79.51700  16091.539  4.941541e-03 0.9960572
mean3           214.16904  95254.098  2.248397e-03 0.9982060
median3        -123.71181  16050.815 -7.707509e-03 0.9938504
std3            -19.02835  62662.598 -3.036637e-04 0.9997577
```

```
one_quartile3   -57.59689  38890.464 -1.481003e-03 0.9988183
th_quartile3    -27.77979  50028.595 -5.552782e-04 0.9995570
min4             29.14211 388504.478  7.501101e-05 0.9999401
max4             30.08913  15591.135  1.929887e-03 0.9984602
mean4         -1407.08972 332716.581 -4.229094e-03 0.9966257
median4        -788.21972 115741.516 -6.810173e-03 0.9945663
std4            968.03914 159691.645  6.061927e-03 0.9951633
one_quartile4   988.07821 128876.692  7.666850e-03 0.9938828
th_quartile4    357.00248 137566.302  2.595130e-03 0.9979294
min5            -31.73682   5639.021 -5.628072e-03 0.9955095
max5            140.27468  19124.133  7.334956e-03 0.9941476
mean5          -637.92548  95921.428 -6.650500e-03 0.9946937
median5          96.81822  38901.444  2.488808e-03 0.9980142
std5           -549.40375  78569.399 -6.992592e-03 0.9944208
one_quartile5   136.71960  21575.458  6.336811e-03 0.9949440
th_quartile5    310.14145  40291.635  7.697416e-03 0.9938584
min6           -690.15394 238930.865 -2.888509e-03 0.9976953
max6             11.81975  28173.484  4.195345e-04 0.9996653
mean6          -778.32178 185390.316 -4.198287e-03 0.9966503
median6        -116.55872 146662.845 -7.947392e-04 0.9993659
std6          -1300.29432 349106.205 -3.724638e-03 0.9970282
one_quartile6   182.57126  83311.130  2.191439e-03 0.9982515
th_quartile6   1320.86339 167516.729  7.884964e-03 0.9937088

p values when for all features when l = 4
                  Estimate   Std. Error       z value  Pr(>|z|)
(Intercept)     140.409767   257870.433  5.444973e-04 0.9995656
min1            -57.782258     8698.569 -6.642731e-03 0.9946999
max1            100.833452    14020.975  7.191615e-03 0.9942620
mean1          -432.871595   150422.115 -2.877712e-03 0.9977039
median1          34.846912    82897.569  4.203611e-04 0.9996646
std1           -638.783997    93178.744 -6.855469e-03 0.9945302
one_quartile1    -9.642309    22843.434 -4.221042e-04 0.9996632
th_quartile1    330.936138    47365.301  6.986890e-03 0.9944253
min2            303.938593 15836155.923  1.919270e-05 0.9999847
max2            -46.523695    21265.796 -2.187724e-03 0.9982545
mean2          1280.760830   442375.457  2.895190e-03 0.9976900
median2        -140.606974   145920.171 -9.635883e-04 0.9992312
std2            278.361410   318607.775  8.736805e-04 0.9993029
one_quartile2  -419.432636   187596.511 -2.235823e-03 0.9982161
th_quartile2   -566.963578   151791.717 -3.735142e-03 0.9970198
min3           -114.301582    13787.871 -8.290010e-03 0.9933856
max3             46.937230     9908.741  4.736952e-03 0.9962205
mean3           274.205983    86919.115  3.154726e-03 0.9974829
median3          48.276813    17567.713  2.748042e-03 0.9978074
std3           -146.361663    54853.616 -2.668223e-03 0.9978711
one_quartile3  -155.883321    24675.617 -6.317302e-03 0.9949596
th_quartile3   -135.792251    57529.166 -2.360407e-03 0.9981167
min4            422.915083 30197625.548  1.400491e-05 0.9999888
max4             57.070410    31464.268  1.813817e-03 0.9985528
mean4         -1586.411268   291172.628 -5.448353e-03 0.9956529
median4        -315.103468   113048.693 -2.787325e-03 0.9977760
std4            225.271588   122291.263  1.842091e-03 0.9985302
```

one_quartile4	612.910776	118658.013	5.165355e-03	0.9958787
th_quartile4	524.583197	113974.790	4.602625e-03	0.9963276
min5	-48.586751	15773.403	-3.080296e-03	0.9975423
max5	114.816522	12649.261	9.076935e-03	0.9927578
mean5	-402.053766	64254.419	-6.257216e-03	0.9950075
median5	93.767552	19427.927	4.826431e-03	0.9961491
std5	-483.209300	100484.254	-4.808806e-03	0.9961631
one_quartile5	29.183659	42688.167	6.836475e-04	0.9994545
th_quartile5	259.787258	36481.239	7.121119e-03	0.9943182
min6	192.989095	15645079.812	1.233545e-05	0.9999902
max6	-2.354802	36015.969	-6.538217e-05	0.9999478
mean6	-453.078494	467751.535	-9.686307e-04	0.9992271
median6	60.340029	87503.363	6.895738e-04	0.9994498
std6	-309.025944	181043.552	-1.706915e-03	0.9986381
one_quartile6	232.525027	155556.402	1.494796e-03	0.9988073
th_quartile6	343.554445	184689.471	1.860173e-03	0.9985158

p values when for all features when l = 5

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	12.61898908	4.5716281	2.7602833828	0.0057751242
min1	-0.20017550	0.4195539	-0.4771151362	0.6332801369
max1	0.31966032	0.4847796	0.6593931221	0.5096433576
mean1	-2.33587928	2.9021966	-0.8048659575	0.4208970349
median1	0.62042993	0.9358734	0.6629421783	0.5073675870
std1	-1.23748849	2.5001206	-0.4949715149	0.6206202278
one_quartile1	0.76363167	0.8558066	0.8922947124	0.3722349877
th_quartile1	0.09541636	1.3360311	0.0714177664	0.9430652702
min2	-41.34723442	7701.2661787	-0.0053688878	0.9957162679
max2	2.23920090	1.1019153	2.0320989818	0.0421436365
mean2	-11.83243726	11.3712186	-1.0405601789	0.2980797207
median2	2.77091537	3.7400040	0.7408856617	0.4587627699
std2	-18.51154338	9.5974012	-1.9288079094	0.0537547136
one_quartile2	-2.23930494	4.0572074	-0.5519325719	0.5809945513
th_quartile2	8.50211151	4.3284914	1.9642205005	0.0495045264
min3	-1.40298339	0.3685695	-3.8065643230	0.0001409107
max3	1.26640423	0.4556227	2.7795022975	0.0054442268
mean3	1.00011652	1.9325027	0.5175239807	0.6047904331
median3	0.88785598	0.9002641	0.9862172258	0.3240265281
std3	-5.06551007	1.5921679	-3.1815176086	0.0014650562
one_quartile3	-2.23173831	0.9204363	-2.4246525914	0.0153230423
th_quartile3	0.14821868	0.7932121	0.1868588155	0.8517713309
min4	-2.38487041	5421.2498394	-0.0004399115	0.9996490014
max4	0.59534425	0.9023913	0.6597406972	0.5094202453
mean4	-2.76802145	10.7480654	-0.2575367146	0.7967644767
median4	-7.47313027	5.0756093	-1.4723612126	0.1409233615
std4	0.52222361	7.0648464	0.0739186080	0.9410751493
one_quartile4	2.54748693	3.6056005	0.7065361131	0.4798548030
th_quartile4	4.16209314	5.0056211	0.8314838616	0.4057003414
min5	-1.27209034	0.3961613	-3.2110413836	0.0013225489
max5	1.74629950	0.5745337	3.0395073239	0.0023696544
mean5	-0.71618862	2.5459420	-0.2813059419	0.7784757533
median5	-0.02837379	1.0327861	-0.0274730579	0.9780824285
std5	-9.07106898	2.5729276	-3.5255827050	0.0004225522

one_quartile5	-1.78695973	1.2558020	-1.4229629230	0.1547469038
th_quartile5	2.70986575	1.1669066	2.3222645418	0.0202186956
min6	-4.50350612	4793.7830004	-0.0009394472	0.9992504297
max6	-0.73692420	0.7989413	-0.9223758829	0.3563325450
mean6	-11.55173474	11.7079336	-0.9866587130	0.3238099767
median6	3.60560721	4.8103548	0.7495512017	0.4535250502
std6	2.00404520	8.3918500	0.2388085113	0.8112540701
one_quartile6	1.89773416	4.2019846	0.4516280637	0.6515369493
th_quartile6	3.74254246	4.1318953	0.9057689430	0.3650581626

p values when for all features when l = 6

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	2.57393348	2.8812087	0.893351979	0.371668710
min1	-0.03804819	0.2483760	-0.153187879	0.878250116
max1	-0.34686221	0.3622869	-0.957424232	0.338353172
mean1	3.04795162	2.0570990	1.481674704	0.138426871
median1	-0.93669285	0.5674171	-1.650801216	0.098779172
std1	0.68593545	1.1604228	0.591108202	0.554447925
one_quartile1	-0.68549130	0.7015796	-0.977068415	0.328535283
th_quartile1	-1.35040606	0.8427316	-1.602415294	0.109063805
min2	-40.39482282	5994.1745542	-0.006739013	0.994623086
max2	2.18048423	0.8395033	2.597350407	0.009394603
mean2	-5.88437360	6.6612440	-0.883374571	0.377033925
median2	-0.81068537	2.4353134	-0.332887491	0.739219212
std2	-15.55648008	7.0889992	-2.194453636	0.028202814
one_quartile2	-3.02874648	2.7287263	-1.109948798	0.267021091
th_quartile2	7.12428939	2.8312955	2.516264864	0.011860601
min3	-0.45018046	0.1990660	-2.261463220	0.023730588
max3	0.47822629	0.2676954	1.786457166	0.074025267
mean3	0.29746312	1.4795747	0.201046369	0.840662317
median3	0.02383371	0.4959706	0.048054678	0.961672666
std3	-1.77786046	1.0531263	-1.688173949	0.091377842
one_quartile3	-0.69107865	0.5193022	-1.330783292	0.183260327
th_quartile3	0.10745479	0.7278365	0.147635896	0.882630125
min4	-18.27967488	5355.3043479	-0.003413377	0.997276524
max4	1.71187716	0.6443649	2.656688981	0.007891221
mean4	15.76906444	7.2876061	2.163819537	0.030478202
median4	-7.73247019	3.3457720	-2.311116872	0.020826400
std4	-11.84973893	5.1328987	-2.308586179	0.020966556
one_quartile4	-7.00336975	2.5941629	-2.699664649	0.006940940
th_quartile4	-2.98239020	2.5409577	-1.173726818	0.240504465
min5	-0.37985617	0.2205133	-1.722600187	0.084960851
max5	0.66834003	0.3575744	1.869093683	0.061609782
mean5	-0.51867474	1.8660650	-0.277951052	0.781049933
median5	-0.08996819	0.6119155	-0.147027136	0.883110602
std5	-2.46348086	1.4096159	-1.747625653	0.080528870
one_quartile5	-0.36220591	0.6416996	-0.564447789	0.572449424
th_quartile5	1.17491420	0.8401405	1.398473470	0.161970934
min6	-38.16801696	4576.0190194	-0.008340878	0.993345019
max6	0.90700766	0.7802523	1.162454304	0.245050976
mean6	3.45648873	7.0094003	0.493121892	0.621926463
median6	-0.17936348	3.2848661	-0.054602981	0.956454764
std6	-12.30323518	6.4582131	-1.905052533	0.056773243

one_quartile6	-3.40344036	3.1168645	-1.091943652	0.274857869
th_quartile6	1.79450584	3.0057819	0.597017984	0.550495374

p values when for all features when l = 7

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	3.64803580	2.4137207	1.511374442	0.1306930774
min1	-0.21048906	0.2524395	-0.833819876	0.4043824931
max1	0.32406797	0.2275744	1.424008968	0.1544438759
mean1	0.28596063	1.3654910	0.209419633	0.8341206673
median1	0.40954415	0.4983880	0.821737592	0.4112262590
std1	-1.00340220	0.9189858	-1.091858216	0.2748954259
one_quartile1	-0.76342764	0.5157892	-1.480115610	0.1388423965
th_quartile1	-0.42483136	0.6671234	-0.636810804	0.5242480876
min2	-40.25621225	4687.4391934	-0.008588103	0.9931477692
max2	1.69307208	0.7385235	2.292509517	0.0218762590
mean2	-8.56940555	6.6952483	-1.279923493	0.2005720445
median2	1.91459939	2.8977877	0.660710729	0.5087978422
std2	-12.43785010	6.1877252	-2.010084420	0.0444222547
one_quartile2	-0.15497790	2.5193572	-0.061514858	0.9509491817
th_quartile2	5.17561679	2.1974830	2.355247718	0.0185103614
min3	-0.57803790	0.1809911	-3.193736107	0.0014044444
max3	0.52402764	0.2272660	2.305789800	0.0211223804
mean3	0.94511562	1.2244777	0.771852062	0.4402020527
median3	0.40846588	0.4630894	0.882045435	0.3777522379
std3	-2.23453497	0.8833742	-2.529545141	0.0114210481
one_quartile3	-1.29192720	0.5476611	-2.358990166	0.0183247413
th_quartile3	-0.17651036	0.4842801	-0.364479911	0.7154996722
min4	-43.40339772	4896.1679589	-0.008864769	0.9929270304
max4	1.09050323	0.4928897	2.212469170	0.0269342642
mean4	5.03288605	6.7595257	0.744562007	0.4565365431
median4	-6.22188082	2.9904454	-2.080586683	0.0374717539
std4	-5.25072111	4.2581811	-1.233090122	0.2175421424
one_quartile4	-0.28395064	2.0653886	-0.137480492	0.8906510111
th_quartile4	-0.01298733	2.2682718	-0.005725648	0.9954316189
min5	-0.69664733	0.1928977	-3.611486490	0.0003044469
max5	1.17031787	0.3100750	3.774305752	0.0001604539
mean5	-1.35638802	1.3501909	-1.004589823	0.3150943997
median5	-0.10761830	0.5500799	-0.195641222	0.8448909936
std5	-4.28216659	1.1071061	-3.867891717	0.0001097804
one_quartile5	-0.18661326	0.6231256	-0.299479365	0.7645743146
th_quartile5	1.63989670	0.5202742	3.151985551	0.0016216430
min6	-32.61520278	3501.2348591	-0.009315343	0.9925675393
max6	-0.01898734	0.6228276	-0.030485705	0.9756796936
mean6	-9.77345942	6.1338978	-1.593352167	0.1110812099
median6	4.79591075	2.7361671	1.752784307	0.0796390378
std6	-0.79270233	4.9932965	-0.158753307	0.8738632380
one_quartile6	3.27768483	2.3855962	1.373947896	0.1694578588
th_quartile6	2.36538090	2.3914203	0.989111315	0.3226086822

p values when for all features when l = 8

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	1.20203724	1.9539911	0.615170278	0.5384422738
min1	-0.41857579	0.1827986	-2.289819085	0.0220318063

max1	0.39130213	0.3036006	1.288871487	0.1974427683
mean1	1.03018620	1.3396873	0.768975108	0.4419080877
median1	-0.45558834	0.4445099	-1.024922882	0.3053995767
std1	-1.89204106	1.1106807	-1.703496900	0.0884751177
one_quartile1	-0.68372351	0.4446250	-1.537753061	0.1241090044
th_quartile1	-0.12085975	0.5849152	-0.206627808	0.8363005414
min2	-36.90960790	4571.6688062	-0.008073552	0.9935583071
max2	0.57984018	0.6351158	0.912967656	0.3612595525
mean2	-8.85632958	5.5593762	-1.593043769	0.1111503725
median2	0.97547284	2.1036347	0.463708288	0.6428567582
std2	-3.48029940	4.8834605	-0.712670741	0.4760495258
one_quartile2	0.15363970	1.9917252	0.077139007	0.9385129625
th_quartile2	4.92636863	2.0565396	2.395465045	0.0165992969
min3	-0.50638087	0.1609862	-3.145492587	0.0016580740
max3	0.42991498	0.1993663	2.156407499	0.0310518596
mean3	0.40187858	1.2187584	0.329744266	0.7415932033
median3	-0.76200347	0.3942891	-1.932600830	0.0532853901
std3	-1.47744141	0.8094803	-1.825172691	0.0679749734
one_quartile3	-0.12185551	0.4372925	-0.278659035	0.7805065033
th_quartile3	0.45931024	0.5682041	0.808354331	0.4188866343
min4	-24.07281959	4009.8699525	-0.006003392	0.9952100153
max4	0.84875240	0.4540431	1.869321154	0.0615781468
mean4	6.08617791	5.6240684	1.082166413	0.2791785897
median4	-6.25272320	2.2334432	-2.799589097	0.0051167694
std4	-5.98684611	3.6787185	-1.627427080	0.1036464241
one_quartile4	-2.25000834	1.8788010	-1.197576705	0.2310818495
th_quartile4	0.47794338	1.8502925	0.258306938	0.7961700319
min5	-0.33819643	0.1636411	-2.066696487	0.0387627624
max5	1.00848460	0.2850153	3.538352365	0.0004026324
mean5	-1.94135116	1.2340495	-1.573154972	0.1156829388
median5	0.47833862	0.4298961	1.112684123	0.2658441167
std5	-2.94607002	0.9403106	-3.133081789	0.0017298123
one_quartile5	0.03542295	0.5040357	0.070278646	0.9439718787
th_quartile5	1.11170220	0.5324899	2.087743124	0.0368210126
min6	1.08097811	4.6071673	0.234629662	0.8144961855
max6	0.31421532	0.5204802	0.603702717	0.5460413072
mean6	-1.56341268	5.7303150	-0.272831889	0.7849824453
median6	2.94640528	2.3860959	1.234822665	0.2168965100
std6	-3.60452477	4.0182782	-0.897032162	0.3697017555
one_quartile6	-1.43734267	2.0013087	-0.718201389	0.4726331214
th_quartile6	0.14868243	2.2629661	0.065702453	0.9476147192

p values when for all features when l = 9

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	1.995914758	1.6388463	1.21787793	0.2232703641
min1	-0.382154888	0.2239815	-1.70618914	0.0879728692
max1	0.602911572	0.3114670	1.93571554	0.0529025504
mean1	-1.452489605	1.5357792	-0.94576720	0.3442673394
median1	-0.359023428	0.4492178	-0.79921905	0.4241634090
std1	-2.562333410	1.1869041	-2.15883784	0.0308627494
one_quartile1	-0.149611739	0.5089787	-0.29394496	0.7687999608
th_quartile1	1.530281536	0.7369926	2.07638657	0.0378582127
min2	-38.071425817	3803.0988314	-0.01001063	0.9920128041

max2	1.331563078	0.6942983	1.91785457	0.0551294544
mean2	-5.122881943	4.8595652	-1.05418524	0.2917981170
median2	-1.799094184	1.9405931	-0.92708470	0.3538825591
std2	-8.516016770	5.0311884	-1.69264518	0.0905230256
one_quartile2	-1.263493745	2.0169740	-0.62643036	0.5310327044
th_quartile2	4.172222267	1.8637107	2.23866410	0.0251777800
min3	-0.564936995	0.1688417	-3.34595711	0.0008199908
max3	0.513180738	0.2047325	2.50659190	0.0121901365
mean3	0.818452078	1.0516725	0.77823857	0.4364283800
median3	-0.009488738	0.3365537	-0.02819383	0.9775075563
std3	-2.136847285	0.7294572	-2.92936626	0.0033965394
one_quartile3	-0.927979299	0.3934811	-2.35838337	0.0183547263
th_quartile3	0.047473762	0.4287094	0.11073645	0.9118253378
min4	-37.250342493	3418.0516424	-0.01089812	0.9913047291
max4	1.276837109	0.4290652	2.97585791	0.0029217020
mean4	7.900145288	4.8282096	1.63624737	0.1017878252
median4	-4.503014275	2.0145089	-2.23529134	0.0253982285
std4	-8.724024619	3.6595049	-2.38393576	0.0171285920
one_quartile4	-2.550086491	1.6891318	-1.50970248	0.1311193579
th_quartile4	-1.261824514	1.6514868	-0.76405367	0.4448352597
min5	-0.248491362	0.1670656	-1.48738760	0.1369124698
max5	0.570976723	0.2662116	2.14482308	0.0319669953
mean5	-0.653328668	1.2553922	-0.52041796	0.6027722948
median5	0.316236149	0.4082157	0.77467901	0.4385293591
std5	-1.886639199	0.8726587	-2.16194389	0.0306225017
one_quartile5	-0.213638256	0.4615489	-0.46287247	0.6434557842
th_quartile5	0.551779519	0.5187260	1.06372063	0.2874552755
min6	1.737585275	6.4970527	0.26744208	0.7891288066
max6	0.725795510	0.5135757	1.41322018	0.1575909942
mean6	-0.762598458	4.8506154	-0.15721685	0.8750739512
median6	1.820393644	1.9507448	0.93317878	0.3507276679
std6	-5.654386461	3.8090958	-1.48444321	0.1376913776
one_quartile6	-2.610808581	1.6811927	-1.55295022	0.1204350268
th_quartile6	0.954187597	1.7436732	0.54722846	0.5842217877

p values when for all features when l = 10

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	3.12525725	1.5233152	2.05161558	4.020704e-02
min1	-0.35359347	0.1402096	-2.52189244	1.167254e-02
max1	0.18139334	0.2193397	0.82699738	4.082385e-01
mean1	0.31263149	0.9513165	0.32863038	7.424351e-01
median1	0.03957648	0.3082987	0.12837059	8.978557e-01
std1	-1.25347953	0.7173331	-1.74741636	8.056514e-02
one_quartile1	-0.45236804	0.3518484	-1.28569020	1.985512e-01
th_quartile1	-0.01155384	0.4049352	-0.02853255	9.772374e-01
min2	-2.34042266	2.3653803	-0.98944879	3.224436e-01
max2	1.04181236	0.5404184	1.92778859	5.388143e-02
mean2	2.91666694	5.0184224	0.58119200	5.611111e-01
median2	-2.42695389	1.8402915	-1.31878773	1.872401e-01
std2	-8.77640147	4.0204819	-2.18292279	2.904150e-02
one_quartile2	-3.37591206	1.8153778	-1.85961959	6.293937e-02
th_quartile2	1.70217330	1.7531683	0.97091265	3.315918e-01
min3	-0.59939228	0.1442275	-4.15588161	3.240356e-05

max3	0.88554225	0.1863187	4.75283626	2.005828e-06
mean3	-1.27975770	0.9803677	-1.30538535	1.917617e-01
median3	0.30192730	0.3101284	0.97355590	3.302771e-01
std3	-2.92056228	0.7097997	-4.11462902	3.878026e-05
one_quartile3	-0.37875621	0.3362215	-1.12650790	2.599506e-01
th_quartile3	0.99289716	0.4274673	2.32274416	2.019290e-02
min4	-31.99121633	2453.7140838	-0.01303787	9.895976e-01
max4	0.75796907	0.4382787	1.72942242	8.373352e-02
mean4	2.22620437	4.2221254	0.52727102	5.980054e-01
median4	-1.61871478	1.7618616	-0.91875251	3.582250e-01
std4	-4.03866338	3.4332285	-1.17634565	2.394568e-01
one_quartile4	-2.53964771	1.5750305	-1.61244346	1.068655e-01
th_quartile4	-0.50299371	1.4071691	-0.35745080	7.207544e-01
min5	-0.44837187	0.1660823	-2.69969654	6.940275e-03
max5	0.53457217	0.2093461	2.55353249	1.066363e-02
mean5	0.87189560	1.1488914	0.75890164	4.479114e-01
median5	0.33034169	0.3870662	0.85345011	3.934097e-01
std5	-1.80712047	0.7741206	-2.33441720	1.957389e-02
one_quartile5	-0.61021564	0.4111032	-1.48433680	1.377196e-01
th_quartile5	-0.35109784	0.5105734	-0.68765397	4.916707e-01
min6	-0.29063887	3.3801319	-0.08598448	9.314788e-01
max6	0.92675260	0.5025808	1.84398734	6.518499e-02
mean6	1.47476637	4.8626164	0.30328659	7.616715e-01
median6	-0.77434405	1.7414290	-0.44466013	6.565654e-01
std6	-8.55639140	3.5993443	-2.37720835	1.744423e-02
one_quartile6	-2.56701711	1.9075571	-1.34570918	1.783963e-01
th_quartile6	1.95197129	1.7107852	1.14097975	2.538783e-01

p values when for all features when l = 11

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	2.09419806	1.3540767	1.54658746	0.1219627547
min1	-0.24379140	0.1678456	-1.45247382	0.1463699040
max1	0.13343097	0.2938260	0.45411557	0.6497456492
mean1	0.94600557	1.2026990	0.78656883	0.4315343147
median1	-0.13616244	0.3859409	-0.35280641	0.7242335762
std1	-0.53699959	0.9783141	-0.54890306	0.5830719760
one_quartile1	-0.46627569	0.3281744	-1.42081676	0.1553700380
th_quartile1	-0.47447619	0.5420800	-0.87528817	0.3814171285
min2	-4.38885098	2.8908605	-1.51818151	0.1289686475
max2	1.80283036	0.5914194	3.04831103	0.0023013160
mean2	-3.77327438	4.4331619	-0.85114744	0.3946874563
median2	-0.82719092	1.5443656	-0.53561859	0.5922221793
std2	-10.30714767	3.7493947	-2.74901643	0.0059774391
one_quartile2	-1.61246601	1.6905063	-0.95383617	0.3401665780
th_quartile2	4.02554823	1.6892155	2.38308739	0.0171681183
min3	-0.52916007	0.1361486	-3.88663605	0.0001016430
max3	0.55240668	0.1654982	3.33784038	0.0008443224
mean3	-1.01209865	0.9494561	-1.06597732	0.2864338937
median3	-0.14130098	0.2868395	-0.49261340	0.6222857795
std3	-2.33177012	0.6966455	-3.34714005	0.0008164995
one_quartile3	-0.08503818	0.3229456	-0.26332048	0.7923035784
th_quartile3	1.12850957	0.4745134	2.37824571	0.0173952304
min4	-32.65508222	1985.6072773	-0.01644589	0.9868786684

```
max4            1.04641264    0.4004969  2.61278558 0.0089807626
mean4           0.67189954    3.8186133  0.17595380 0.8603302358
median4        -2.17916217    1.5133191 -1.43998857 0.1498706326
std4           -4.93532303    2.8911393 -1.70705129 0.0878125196
one_quartile4  -1.16359316    1.4362936 -0.81013601 0.4178620114
th_quartile4    0.53789152    1.3160202  0.40872588 0.6827408398
min5           -0.22718254    0.1561422 -1.45497181 0.1456770653
max5            0.72224201    0.2461096  2.93463524 0.0033393999
mean5          -1.42929270    1.2797043 -1.11689295 0.2640400979
median5         0.17922332    0.3983397  0.44992580 0.6527639418
std5           -1.65256415    0.8675678 -1.90482431 0.0568029131
one_quartile5   0.42253617    0.4108218  1.02851443 0.3037079060
th_quartile5    0.66582345    0.5400545  1.23288196 0.2176198056
min6           -3.87244127    2.8265523 -1.37002285 0.1706797702
max6            0.37187191    0.4735026  0.78536401 0.4322401746
mean6          -0.26903727    3.9416267 -0.06825539 0.9455823353
median6         0.12535598    1.4423600  0.08691033 0.9307427868
std6           -3.50857283    3.0356593 -1.15578609 0.2477686642
one_quartile6   0.89666276    1.6519338  0.54279581 0.5872704012
th_quartile6    0.26823799    1.4156008  0.18948703 0.8497111172

p values when for all features when l = 12
                   Estimate   Std. Error     z value     Pr(>|z|)
(Intercept)     3.522117584    1.2241486  2.87719771 0.0040122412
min1           -0.259406992    0.1414328 -1.83413544 0.0666338925
max1            0.012403738    0.2024010  0.06128299 0.9511338383
mean1           0.334534710    0.8880594  0.37670308 0.7063942768
median1        -0.355201044    0.3250208 -1.09285637 0.2744568681
std1           -0.787435444    0.7754312 -1.01548070 0.3098767441
one_quartile1  -0.254861205    0.3141278 -0.81132978 0.4171763105
th_quartile1    0.261589656    0.4301083  0.60819483 0.5430582644
min2           -0.844784363    1.9848516 -0.42561588 0.6703877663
max2            1.299337874    0.5895023  2.20412693 0.0275154190
mean2          -4.659641426    3.9947329 -1.16644630 0.2434340476
median2         1.285600932    1.4116741  0.91069241 0.3624574645
std2           -6.714384740    3.4904252 -1.92365809 0.0543974547
one_quartile2  -1.563281119    1.5549978 -1.00532692 0.3147394578
th_quartile2    2.474668505    1.4667096  1.68722456 0.0915601798
min3           -0.493331162    0.1383102 -3.56684682 0.0003613025
max3            0.498127467    0.1656167  3.00771287 0.0026322171
mean3           0.395385199    0.8307767  0.47592235 0.6341297007
median3         0.107202753    0.2580153  0.41548994 0.6777832783
std3           -1.642749627    0.5828700 -2.81838074 0.0048266538
one_quartile3  -0.577868950    0.2887003 -2.00162261 0.0453253352
th_quartile3   -0.004304513    0.3489054 -0.01233719 0.9901565944
min4          -31.828528856 1807.4294460 -0.01760983 0.9859501131
max4            1.167151612    0.3935438  2.96574788 0.0030194791
mean4           3.848900422    3.8835366  0.99108127 0.3216458979
median4        -1.137787297    1.4789609 -0.76931532 0.4417061450
std4           -6.762959117    2.7983694 -2.41674992 0.0156597721
one_quartile4  -3.649776609    1.4338417 -2.54545291 0.0109136082
th_quartile4   -0.793806422    1.2621857 -0.62891411 0.5294052874
min5           -0.189113689    0.1406282 -1.34477775 0.1786970190
```

max5	0.437413128	0.2150976	2.03355658	0.0419963239
mean5	-0.245887089	1.1533293	-0.21319765	0.8311728026
median5	-0.068838853	0.3706220	-0.18573869	0.8526496832
std5	-0.612516220	0.6895325	-0.88830659	0.3743758582
one_quartile5	0.292493298	0.3585288	0.81581531	0.4146057783
th_quartile5	0.071721487	0.4684801	0.15309399	0.8783241517
min6	-4.883146697	2.4315068	-2.00828009	0.0446135367
max6	0.523730146	0.4436611	1.18047342	0.2378119778
mean6	2.376064737	3.6006099	0.65990619	0.5093140342
median6	-0.187628026	1.3846604	-0.13550473	0.8922128263
std6	-5.625070093	2.9610406	-1.89969366	0.0574733329
one_quartile6	-1.931300703	1.5962896	-1.20986860	0.2263293171
th_quartile6	0.228500587	1.3463021	0.16972460	0.8652267230

p values when for all features when l = 13

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	2.91224490	1.2102071	2.40640218	1.611052e-02
min1	-0.29271332	0.1573683	-1.86005314	6.287801e-02
max1	0.32497116	0.2693070	1.20669417	2.275499e-01
mean1	0.59937037	1.0718265	0.55920465	5.760221e-01
median1	-0.15373491	0.3645695	-0.42168885	6.732521e-01
std1	-1.26195397	0.8523221	-1.48060685	1.387114e-01
one_quartile1	-0.63116745	0.3507700	-1.79937704	7.195906e-02
th_quartile1	-0.09808047	0.5012321	-0.19567875	8.448616e-01
min2	-3.97469018	2.2179567	-1.79205043	7.312489e-02
max2	0.71038725	0.5441698	1.30545136	1.917392e-01
mean2	-2.14308010	4.2800635	-0.50071222	6.165737e-01
median2	-0.53798314	1.6138563	-0.33335256	7.388682e-01
std2	-4.30915249	3.3194446	-1.29815466	1.942342e-01
one_quartile2	-0.49864927	1.6099030	-0.30973871	7.567597e-01
th_quartile2	1.65983655	1.4030235	1.18304258	2.367923e-01
min3	-0.61243281	0.1381234	-4.43395303	9.252079e-06
max3	0.74627638	0.1816434	4.10846904	3.982906e-05
mean3	-0.63594498	0.9376685	-0.67821942	4.976326e-01
median3	0.23330840	0.2588153	0.90144757	3.673504e-01
std3	-3.05593924	0.6575534	-4.64743910	3.360813e-06
one_quartile3	-0.66952216	0.3226633	-2.07498683	3.798776e-02
th_quartile3	0.87481111	0.3970150	2.20347097	2.756157e-02
min4	-35.57793304	1749.3936762	-0.02033729	9.837743e-01
max4	1.38210741	0.4225180	3.27112078	1.071221e-03
mean4	-4.06838464	3.7549727	-1.08346584	2.786017e-01
median4	0.19230980	1.6115307	0.11933362	9.050110e-01
std4	-4.16062755	2.7625629	-1.50607524	1.320479e-01
one_quartile4	-0.60481162	1.4491858	-0.41734580	6.764255e-01
th_quartile4	1.32635624	1.2030003	1.10254027	2.702269e-01
min5	-0.38272959	0.1555479	-2.46052486	1.387340e-02
max5	0.40430838	0.2121960	1.90535326	5.673417e-02
mean5	0.69511958	1.1386046	0.61050129	5.415298e-01
median5	0.01870752	0.3288786	0.05688276	9.546386e-01
std5	-1.95219369	0.7477266	-2.61083903	9.032040e-03
one_quartile5	-0.77045382	0.3825066	-2.01422346	4.398608e-02
th_quartile5	0.33045196	0.4598082	0.71867343	4.723422e-01
min6	0.35974669	1.8927653	0.19006408	8.492589e-01

max6	0.59702628	0.5057153	1.18055798	2.377784e-01
mean6	-1.51818757	3.7535035	-0.40447213	6.858656e-01
median6	0.42294296	1.4644833	0.28880014	7.727343e-01
std6	-3.91188406	2.9884115	-1.30901789	1.905283e-01
one_quartile6	-1.72078310	1.4584317	-1.17988596	2.380456e-01
th_quartile6	1.72140336	1.3666823	1.25954902	2.078321e-01

p values when for all features when l = 14

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	2.82182647	1.1338297	2.48875697	1.281906e-02
min1	-0.30757564	0.1787422	-1.72077771	8.529117e-02
max1	0.23543607	0.2428035	0.96965685	3.322176e-01
mean1	0.20771002	0.9588522	0.21662361	8.285017e-01
median1	-0.22594638	0.2985407	-0.75683604	4.491481e-01
std1	-1.43212174	0.8299158	-1.72562293	8.441527e-02
one_quartile1	-0.42227056	0.3405990	-1.23978808	2.150538e-01
th_quartile1	0.26001885	0.4316036	0.60244825	5.468758e-01
min2	-4.01669053	1.9402462	-2.07019630	3.843397e-02
max2	1.78894170	0.5913738	3.02506084	2.485831e-03
mean2	-3.83231492	3.9633302	-0.96694313	3.335725e-01
median2	-0.28415413	1.3248182	-0.21448537	8.301686e-01
std2	-9.76619007	3.3886239	-2.88205197	3.950946e-03
one_quartile2	-1.38114783	1.5214572	-0.90777960	3.639947e-01
th_quartile2	3.78181409	1.5326671	2.46747258	1.360706e-02
min3	-0.53134779	0.1259301	-4.21938810	2.449663e-05
max3	0.56255122	0.1661692	3.38541229	7.107142e-04
mean3	0.49763456	0.7791942	0.63865282	5.230488e-01
median3	-0.04923073	0.2278267	-0.21608844	8.289188e-01
std3	-1.61357026	0.5005460	-3.22362039	1.265811e-03
one_quartile3	-0.54590883	0.2709378	-2.01488622	4.391657e-02
th_quartile3	0.02630419	0.3282557	0.08013325	9.361313e-01
min4	-4.80906307	3.5867794	-1.34077470	1.799936e-01
max4	1.05999496	0.3901462	2.71691747	6.589304e-03
mean4	2.36878848	3.1918932	0.74212648	4.580107e-01
median4	-1.49556133	1.2220502	-1.22381333	2.210227e-01
std4	-6.47086076	2.5253538	-2.56235806	1.039641e-02
one_quartile4	-2.37309632	1.2590236	-1.88487046	5.944733e-02
th_quartile4	0.16620294	1.0606315	0.15670186	8.754798e-01
min5	-0.19506431	0.1584658	-1.23095557	2.183395e-01
max5	1.02663177	0.2250625	4.56153988	5.077983e-06
mean5	-2.45444142	1.0163442	-2.41497074	1.573647e-02
median5	0.85221423	0.3276590	2.60091850	9.297454e-03
std5	-1.98286515	0.7211637	-2.74953536	5.967982e-03
one_quartile5	0.33276443	0.3538886	0.94030829	3.470594e-01
th_quartile5	0.73460028	0.3868128	1.89911048	5.754995e-02
min6	-2.10083816	1.6555596	-1.26895955	2.044555e-01
max6	0.99298452	0.4865651	2.04080523	4.127019e-02
mean6	-1.17596517	3.5579970	-0.33051326	7.410122e-01
median6	-0.26186112	1.3907303	-0.18829037	8.506490e-01
std6	-6.14324655	2.7380399	-2.24366586	2.485391e-02
one_quartile6	-0.91631051	1.4125626	-0.64868664	5.165409e-01
th_quartile6	1.47181337	1.2625074	1.16578596	2.437010e-01

p values when for all features when l = 15

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	2.904352198	1.0446292	2.78027097	5.431356e-03
min1	-0.411919831	0.1969360	-2.09164305	3.647046e-02
max1	0.680908569	0.3122953	2.18033530	2.923262e-02
mean1	-1.058166398	1.2129594	-0.87238402	3.829989e-01
median1	0.147288763	0.3361827	0.43812114	6.612985e-01
std1	-2.279160698	0.9697699	-2.35020767	1.876294e-02
one_quartile1	-0.330586323	0.3521790	-0.93868833	3.478908e-01
th_quartile1	0.736735034	0.5405784	1.36286436	1.729253e-01
min2	-3.121065004	1.7590937	-1.77424605	7.602246e-02
max2	1.709936289	0.6069138	2.81742841	4.840991e-03
mean2	-0.870630234	3.9737476	-0.21909550	8.265757e-01
median2	0.469749129	1.3404571	0.35043950	7.260089e-01
std2	-9.324912376	3.3986392	-2.74371945	6.074743e-03
one_quartile2	-3.555086917	1.5562857	-2.28434082	2.235151e-02
th_quartile2	1.401824109	1.4168721	0.98937941	3.224775e-01
min3	-0.549674331	0.1241606	-4.42712453	9.549760e-06
max3	0.451533081	0.1723475	2.61989937	8.795572e-03
mean3	0.487085940	0.8281229	0.58818075	5.564110e-01
median3	0.009976968	0.2410543	0.04138888	9.669859e-01
std3	-2.092810335	0.5648522	-3.70505836	2.113421e-04
one_quartile3	-0.748269778	0.2772598	-2.69880349	6.958925e-03
th_quartile3	0.259465804	0.3401333	0.76283560	4.455614e-01
min4	-33.914909807	1459.1517652	-0.02324289	9.814565e-01
max4	1.359058962	0.3577456	3.79895379	1.453082e-04
mean4	3.156809125	3.3986331	0.92884669	3.529685e-01
median4	-1.092107643	1.3510726	-0.80832641	4.189027e-01
std4	-7.653690528	2.5551994	-2.99533979	2.741393e-03
one_quartile4	-4.120191170	1.3675757	-3.01277000	2.588750e-03
th_quartile4	0.322302720	1.0065264	0.32021287	7.488070e-01
min5	-0.141379064	0.1488449	-0.94984147	3.421928e-01
max5	0.569102877	0.2169124	2.62365357	8.699222e-03
mean5	-0.586522076	1.0288161	-0.57009417	5.686138e-01
median5	0.537012912	0.2885671	1.86096373	6.274930e-02
std5	-0.778070096	0.6723665	-1.15721124	2.471861e-01
one_quartile5	-0.043670822	0.3439120	-0.12698254	8.989542e-01
th_quartile5	-0.014093202	0.4135818	-0.03407598	9.728166e-01
min6	-3.467956449	1.6201899	-2.14046289	3.231738e-02
max6	1.047618146	0.4586592	2.28408832	2.236634e-02
mean6	10.009879542	3.4874954	2.87022016	4.101861e-03
median6	-2.500967435	1.3340445	-1.87472560	6.083048e-02
std6	-9.892313789	2.8053354	-3.52624997	4.214889e-04
one_quartile6	-4.469886105	1.5077036	-2.96469822	3.029800e-03
th_quartile6	-2.178699470	1.2692939	-1.71646569	8.607684e-02

p values when for all features when l = 16

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	3.39982481	1.0095284	3.36773557	7.578825e-04
min1	-0.39691592	0.1627760	-2.43841785	1.475171e-02
max1	0.23804543	0.2872731	0.82863818	4.073092e-01
mean1	0.69586120	1.0077426	0.69051484	4.898705e-01
median1	-0.10655118	0.2939041	-0.36253720	7.169506e-01

```
std1            -1.50508890    0.8812654 -1.70787240 8.766002e-02
one_quartile1   -0.65238261    0.3258890 -2.00185518 4.530031e-02
th_quartile1    -0.02835256    0.4400501 -0.06443030 9.486276e-01
min2            -3.71521882    1.6198234 -2.29359504 2.181377e-02
max2             0.44107631    0.5345281  0.82516954 4.092753e-01
mean2            4.95184907    3.8872209  1.27387899 2.027063e-01
median2         -0.50296172    1.1431376 -0.43998355 6.599490e-01
std2            -3.55657423    2.9089314 -1.22263944 2.214659e-01
one_quartile2   -2.84189705    1.5137704 -1.87736338 6.046831e-02
th_quartile2    -1.66019070    1.4114003 -1.17627207 2.394862e-01
min3            -0.53060043    0.1324082 -4.00730855 6.141461e-05
max3             0.73013305    0.1677028  4.35373230 1.338391e-05
mean3           -1.32880086    0.8368872 -1.58778977 1.123339e-01
median3          0.48519334    0.2351768  2.06310043 3.910309e-02
std3            -2.52636409    0.5825632 -4.33663491 1.446806e-05
one_quartile3   -0.30349798    0.2780099 -1.09168031 2.749736e-01
th_quartile3     0.89080198    0.3621878  2.45950284 1.391296e-02
min4           -32.36721292 1443.5419018 -0.02242208 9.821113e-01
max4             1.35419680    0.3466036  3.90704746 9.343081e-05
mean4           -1.69837802    3.1627077 -0.53700127 5.912667e-01
median4         -0.63026694    1.2081333 -0.52168657 6.018886e-01
std4            -5.40486055    2.2409551 -2.41185578 1.587156e-02
one_quartile4   -1.11225120    1.2249194 -0.90801991 3.638677e-01
th_quartile4     1.01477979    0.9572430  1.06010677 2.890960e-01
min5            -0.10517219    0.1613127 -0.65197725 5.144158e-01
max5             0.73206580    0.2077232  3.52423697 4.247042e-04
mean5           -1.78549436    1.0452707 -1.70816449 8.760583e-02
median5          0.44439348    0.2853926  1.55713048 1.194395e-01
std5            -1.16971846    0.6439345 -1.81651764 6.929100e-02
one_quartile5    0.47818274    0.3676334  1.30070552 1.933593e-01
th_quartile5     0.51907458    0.3944179  1.31605229 1.881565e-01
min6            -1.37568382    1.2692798 -1.08383026 2.784401e-01
max6             0.93292705    0.4363998  2.13778071 3.253455e-02
mean6            1.88393113    3.3948089  0.55494468 5.789325e-01
median6         -0.77733517    1.1988025 -0.64842636 5.167092e-01
std6            -6.90882630    2.5269503 -2.73405709 6.255920e-03
one_quartile6   -2.27033501    1.3921067 -1.63086281 1.029193e-01
th_quartile6     0.77666699    1.1249230  0.69041789 4.899314e-01

p values when for all features when l = 17
                 Estimate Std. Error     z value     Pr(>|z|)
(Intercept)    3.15378369  0.9589294  3.28885924 0.0010059432
min1          -0.41541244  0.1582195 -2.62554507 0.0086510355
max1           0.57073083  0.2731196  2.08967399 0.0366470955
mean1         -1.19232341  0.9053031 -1.31704332 0.1878241034
median1        0.20411738  0.2747437  0.74293737 0.4575195969
std1          -2.18397822  0.8360503 -2.61225703 0.0089946602
one_quartile1 -0.16569844  0.2938132 -0.56395855 0.5727823435
th_quartile1   0.75086324  0.4268429  1.75910920 0.0785589623
min2          -2.21753814  1.5658057 -1.41622815 0.1567087175
max2           0.49855251  0.6026720  0.82723683 0.4081028424
mean2          1.36776096  4.0466855  0.33799537 0.7353666797
median2       -0.21303928  1.2770264 -0.16682449 0.8675081513
```

```
std2            -4.30500600  3.1576550 -1.36335540 0.1727705311
one_quartile2   -2.37006144  1.5147918 -1.56461198 0.1176739185
th_quartile2     0.07412798  1.3912319  0.05328226 0.9575070131
min3            -0.50570985  0.1305832 -3.87270361 0.0001076347
max3             0.56862034  0.1668100  3.40879062 0.0006525154
mean3           -0.01382535  0.7815411 -0.01768986 0.9858862684
median3          0.24158715  0.2323803  1.03961987 0.2985165409
std3            -1.98143237  0.5318627 -3.72545843 0.0001949606
one_quartile3   -0.57378502  0.2601748 -2.20538274 0.0274272485
th_quartile3     0.23646575  0.3021460  0.78262086 0.4338497899
min4            -3.85015117  3.4948124 -1.10167607 0.2706025254
max4             1.15675564  0.3635767  3.18159977 0.0014646407
mean4           -3.63875957  3.2263204 -1.12783577 0.2593892850
median4          0.60149902  1.2654920  0.47530842 0.6345671608
std4            -2.90794326  2.3533445 -1.23566409 0.2165834525
one_quartile4   -0.15775285  1.2313219 -0.12811666 0.8980566552
th_quartile4     0.67911250  0.9707736  0.69955803 0.4842033620
min5            -0.28804651  0.1463273 -1.96850900 0.0490095014
max5             0.73013713  0.2281708  3.19995858 0.0013744734
mean5           -0.85842709  1.0535662 -0.81478230 0.4151969332
median5         -0.16816026  0.3071936 -0.54740800 0.5840984593
std5            -1.45560288  0.6788067 -2.14435537 0.0320044237
one_quartile5    0.35002727  0.3442412  1.01680811 0.3092447240
th_quartile5     0.51057354  0.3970910  1.28578464 0.1985182328
min6            -1.74189258  1.2991318 -1.34081284 0.1799812261
max6             0.57091512  0.4596862  1.24196717 0.2142486739
mean6            0.73206910  3.2170459  0.22755942 0.8199887730
median6         -0.07254759  1.2168061 -0.05962132 0.9524572374
std6            -4.86755472  2.5904271 -1.87905488 0.0602369998
one_quartile6   -1.70723070  1.3036590 -1.30956843 0.1903418744
th_quartile6     0.87500768  1.0253049  0.85341219 0.3934307653
```

p values when for all features when l = 18

```
                   Estimate Std. Error     z value     Pr(>|z|)
(Intercept)      2.78613624  0.9034177  3.08399568 0.0020424056
min1            -0.21019455  0.1329442 -1.58107303 0.1138613415
max1             0.28784558  0.2671138  1.07761405 0.2812060286
mean1            0.02765776  0.8866222  0.03119453 0.9751144034
median1          0.04664678  0.2652142  0.17588339 0.8603855521
std1            -0.77733762  0.7855621 -0.98953044 0.3224036859
one_quartile1   -0.23221947  0.2589497 -0.89677439 0.3698393154
th_quartile1    -0.13105700  0.4238426 -0.30921149 0.7571606532
min2            -2.52926195  1.3254012 -1.90829910 0.0563525696
max2             0.87068563  0.5000189  1.74130548 0.0816300459
mean2            1.74689112  3.2361567  0.53980424 0.5893320400
median2         -1.81663616  1.1014773 -1.64927247 0.0990918267
std2            -5.66719115  2.6950556 -2.10281044 0.0354823429
one_quartile2   -2.26177329  1.2794427 -1.76778000 0.0770976894
th_quartile2     1.02964461  1.1294534  0.91163091 0.3619630469
min3            -0.49866583  0.1201120 -4.15167344 0.0000330053
max3             0.60350006  0.1609498  3.74961668 0.0001771051
mean3           -0.25697481  0.7172710 -0.35826740 0.7201432122
median3          0.16894689  0.2271864  0.74364869 0.4570890390
```

std3	-1.90886550	0.5211784	-3.66259506	0.0002496731
one_quartile3	-0.37000835	0.2151776	-1.71954898	0.0855144569
th_quartile3	0.32253862	0.2893898	1.11454736	0.2650444371
min4	-4.82808485	2.7822029	-1.73534606	0.0826795284
max4	1.19962363	0.3746075	3.20234828	0.0013631209
mean4	4.00799862	2.6767094	1.49736038	0.1342995115
median4	-2.81492617	1.0866045	-2.59057094	0.0095816869
std4	-6.90342080	2.2938142	-3.00958151	0.0026160787
one_quartile4	-1.70781364	1.0555303	-1.61796746	0.1056696099
th_quartile4	-0.11056864	0.8057500	-0.13722450	0.8908533481
min5	0.04379656	0.1526507	0.28690711	0.7741834490
max5	0.69524107	0.2207096	3.15002691	0.0016325543
mean5	-2.47050193	0.8978573	-2.75155287	0.0059313445
median5	0.75632845	0.2747474	2.75281357	0.0059085532
std5	-1.00545491	0.6243634	-1.61036813	0.1073175145
one_quartile5	0.54028130	0.3002087	1.79968589	0.0719102509
th_quartile5	0.67225214	0.3248124	2.06966296	0.0384839178
min6	-0.85913518	1.1486059	-0.74798080	0.4544717410
max6	0.44295274	0.4131915	1.07202783	0.2837075308
mean6	0.98087913	2.8351463	0.34597125	0.7293643179
median6	-0.34527765	0.9580512	-0.36039582	0.7185511526
std6	-3.13541230	2.2268505	-1.40800305	0.1591301665
one_quartile6	-0.98833875	1.1524864	-0.85757086	0.3911294715
th_quartile6	0.02836561	0.9618938	0.02948934	0.9764743208

p values when for all features when l = 19

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	3.45548547	0.8968469	3.8529266	1.167144e-04
min1	-0.30877975	0.1498441	-2.0606737	3.933418e-02
max1	0.55771328	0.2763215	2.0183491	4.355492e-02
mean1	-1.13837041	0.8765226	-1.2987348	1.940350e-01
median1	-0.26229198	0.2762285	-0.9495473	3.423424e-01
std1	-1.65024206	0.7435097	-2.2195301	2.645069e-02
one_quartile1	0.07773151	0.2602111	0.2987248	7.651500e-01
th_quartile1	0.82888415	0.3960745	2.0927482	3.637164e-02
min2	-3.19965157	1.3510619	-2.3682494	1.787249e-02
max2	0.61588463	0.5190295	1.1866082	2.353822e-01
mean2	4.48804488	3.1780517	1.4122001	1.578911e-01
median2	-0.97763742	1.0443863	-0.9360880	3.492279e-01
std2	-5.50580396	2.6663857	-2.0648940	3.893303e-02
one_quartile2	-3.29233851	1.2675468	-2.5974097	9.392981e-03
th_quartile2	-0.95005319	1.1402310	-0.8332112	4.047256e-01
min3	-0.49805191	0.1220865	-4.0795000	4.513267e-05
max3	0.56033092	0.1544737	3.6273542	2.863404e-04
mean3	-0.64155707	0.7179909	-0.8935449	3.715654e-01
median3	0.14794067	0.2075016	0.7129617	4.758694e-01
std3	-1.98241339	0.5040688	-3.9328227	8.395415e-05
one_quartile3	-0.24113634	0.2309417	-1.0441436	2.964189e-01
th_quartile3	0.61499208	0.3005490	2.0462292	4.073382e-02
min4	-3.53498741	2.2948154	-1.5404235	1.234572e-01
max4	1.53805583	0.3451323	4.4564238	8.333822e-06
mean4	-0.46191756	2.4603731	-0.1877429	8.510782e-01
median4	-0.33519045	0.9951374	-0.3368283	7.362463e-01

```
std4          -6.57693954  2.0081920 -3.2750551 1.056414e-03
one_quartile4 -2.08048468  0.9956236 -2.0896296 3.665108e-02
th_quartile4   0.98560101  0.7608723  1.2953567 1.951972e-01
min5          -0.33431769  0.1426079 -2.3443135 1.906214e-02
max5           0.53814346  0.2125941  2.5313193 1.136344e-02
mean5          0.33765735  0.9222185  0.3661360 7.142636e-01
median5        0.17642757  0.2788659  0.6326611 5.269550e-01
std5          -1.48169357  0.6376608 -2.3236393 2.014484e-02
one_quartile5 -0.38826934  0.3032181 -1.2804951 2.003711e-01
th_quartile5  -0.04550321  0.3656262 -0.1244528 9.009568e-01
min6          -2.54389906  1.1819625 -2.1522671 3.137633e-02
max6           1.03672516  0.4335977  2.3909837 1.680330e-02
mean6          0.86549868  2.8018034  0.3089077 7.573917e-01
median6       -0.26125155  1.0402582 -0.2511411 8.017050e-01
std6          -6.49851850  2.3910281 -2.7178763 6.570240e-03
one_quartile6 -1.81634010  1.1658443 -1.5579611 1.192425e-01
th_quartile6   0.88235612  0.9656019  0.9137887 3.608279e-01

p values when for all features when l = 20
                 Estimate Std. Error    z value     Pr(>|z|)
(Intercept)    3.15789080  0.8564960  3.6869883 2.269238e-04
min1          -0.59349891  0.2418174 -2.4543270 1.411485e-02
max1           0.86374044  0.3007262  2.8721820 4.076481e-03
mean1         -1.20725992  1.1905338 -1.0140493 3.105592e-01
median1        0.12667870  0.3383544  0.3743965 7.081093e-01
std1          -2.94077313  0.9617952 -3.0575876 2.231264e-03
one_quartile1 -0.34961042  0.3624966 -0.9644516 3.348196e-01
th_quartile1   0.93556506  0.4803269  1.9477672 5.144282e-02
min2          -1.78849141  1.2505777 -1.4301322 1.526791e-01
max2           1.03111895  0.5927190  1.7396422 8.192186e-02
mean2          1.99063618  3.8785456  0.5132430 6.077814e-01
median2       -2.26759035  1.2828202 -1.7676603 7.711772e-02
std2          -6.32748221  2.8506652 -2.2196512 2.644246e-02
one_quartile2 -2.54485520  1.4627267 -1.7398023 8.189374e-02
th_quartile2   1.11168020  1.2218164  0.9098586 3.628971e-01
min3          -0.66568069  0.1354952 -4.9129453 8.971833e-07
max3           0.93422352  0.1808918  5.1645421 2.410282e-07
mean3         -1.64920383  0.8416012 -1.9596026 5.004226e-02
median3        0.53423513  0.2462465  2.1695139 3.004369e-02
std3          -3.23494857  0.5845175 -5.5343916 3.123106e-08
one_quartile3 -0.38563634  0.2405462 -1.6031695 1.088972e-01
th_quartile3   1.18334386  0.3502526  3.3785438 7.287083e-04
min4          -5.05053824  2.2688624 -2.2260222 2.601269e-02
max4           1.58806206  0.3919960  4.0512201 5.095125e-05
mean4          3.07946843  2.7344670  1.1261677 2.600945e-01
median4       -2.12427189  1.0951620 -1.9396874 5.241769e-02
std4          -8.68117526  2.2690371 -3.8259293 1.302797e-04
one_quartile4 -3.02834322  1.1348971 -2.6683856 7.621672e-03
th_quartile4   0.27030495  0.8314504  0.3251005 7.451050e-01
min5          -0.02318920  0.1543870 -0.1502018 8.806054e-01
max5           0.27474047  0.2196353  1.2508939 2.109732e-01
mean5          0.17765522  0.9463956  0.1877177 8.510979e-01
median5        0.03284379  0.3022085  0.1086792 9.134569e-01
```

```
std5           0.11113492  0.6540200  0.1699259 8.650684e-01
one_quartile5  0.17308849  0.3080325  0.5619163 5.741730e-01
th_quartile5  -0.36550971  0.3463258 -1.0553928 2.912457e-01
min6          -2.03074430  1.1026779 -1.8416478 6.552668e-02
max6           0.41482600  0.4417317  0.9390904 3.476843e-01
mean6          6.57153319  3.2833001  2.0015025 4.533827e-02
median6       -0.89868301  1.1036720 -0.8142664 4.154924e-01
std6          -5.67366901  2.3259591 -2.4392815 1.471650e-02
one_quartile6 -3.03514166  1.3062496 -2.3235541 2.014940e-02
th_quartile6  -1.52913766  1.0653351 -1.4353584 1.511850e-01
```

In [39]:

```
#index_2=[0,1,2,7,8,9,35,36,37]
for i in range(20):
    i=i+1
    createVar['pse_9'+str(i)] = createVar['dataset_'+str(i)][:,index_2]
    createVar['pse_new_dataset_'+str(i)]=np.column_stack((createVar['pse_9'+str(
i)],createVar['y_'+str(i)]))
    createVar['pse_df_bi_'+str(i)] = pd.DataFrame(createVar['pse_new_dataset_'+s
tr(i)],
                                                  columns = ['min1','max1','mean
1','min2','max2','mean2',
                                                             'min6','max6','mean
6','label'])
```

In [40]:

```
pse_acc_9 = list()
for i in range(20):
    i=i+1
    createVar['pse_clf_'+str(i)] = LogisticRegression(random_state = 0)
    createVar['pse_clf_'+str(i)].fit( createVar['pse_9'+str(i)],createVar['y_'+s
tr(i)].ravel())
    createVar['pse_y_pred_'+str(i)]= createVar['pse_clf_'+str(i)].predict(create
Var['pse_9'+str(i)])
    createVar['pse_test_acc_'+str(i)]=accuracy_score(createVar['y_'+str(i)],crea
teVar['pse_y_pred_'+str(i)])
    pse_acc_9.append(createVar['pse_test_acc_'+str(i)])
```

In [41]:

```
for i in range (20):
    i=i+1
    print("use the selected 9 features in d)i, the predicted class for training
set when l = "+str(i)+"is:")
    print( createVar['pse_y_pred_'+str(i)])
    print( "test accuracy when l = "+str(i)+"is:")
    print( createVar['pse_test_acc_'+str(i)])
```

```
use the selected 9 features in d)i, the predicted class for training
set when l = 1is:
```

```
[0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
test accuracy when l = 1is:
0.8840579710144928
use the selected 9 features in d)i, the predicted class for training
set when l = 2is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 1. 1. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
test accuracy when l = 2is:
0.8840579710144928
use the selected 9 features in d)i, the predicted class for training
set when l = 3is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1
. 0.
 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
test accuracy when l = 3is:
0.8695652173913043
use the selected 9 features in d)i, the predicted class for training
set when l = 4is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
```

```
 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0
. 0.
 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
test accuracy when l = 4is:
0.8586956521739131
use the selected 9 features in d)i, the predicted class for training
set when l = 5is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 1. 0. 0. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0.]
test accuracy when l = 5is:
0.8724637681159421
use the selected 9 features in d)i, the predicted class for training
set when l = 6is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
```

```
. 0.
 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1
. 0.
 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0.]
test accuracy when l = 6is:
0.8695652173913043
use the selected 9 features in d)i, the predicted class for training
set when l = 7is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 1. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
```

```
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 1. 0. 0. 1. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0.]
test accuracy when l = 7is:
0.8633540372670807
use the selected 9 features in d)i, the predicted class for training
set when l = 8is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0
```

```
. 0.
 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.]
test accuracy when l = 8is:
0.8659420289855072
use the selected 9 features in d)i, the predicted class for training
set when l = 9is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 0. 1. 1. 0
. 0.
 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
. 0.
 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
```

```
 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
test accuracy when l = 9is:
0.8743961352657005
use the selected 9 features in d)i, the predicted class for training
set when l = 10is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 1. 1. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
```

```
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
test accuracy when l = 10is:
0.8695652173913043
use the selected 9 features in d)i, the predicted class for training
set when l = 11is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
```

```
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
test accuracy when l = 11is:
0.8669301712779973
use the selected 9 features in d)i, the predicted class for training
set when l = 12is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
```

```
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
```

```
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
test accuracy when l = 12is:
0.8683574879227053
use the selected 9 features in d)i, the predicted class for training
set when l = 13is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0
. 0.
 1. 0. 1. 1. 1. 1. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
```

```
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0.]
test accuracy when l = 13is:
0.8706800445930881
use the selected 9 features in d)i, the predicted class for training
set when l = 14is:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 1. 0. 1. 1. 0. 0. 0. 0. 0. 1. 0. 0. 0
```

```
. 0.
 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 1.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
```

```
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0.]
test accuracy when l = 14is:
0.8726708074534162
use the selected 9 features in d)i, the predicted class for training
set when l = 15is:
[0. 0. 0. ... 0. 0. 0.]
test accuracy when l = 15is:
0.8705314009661835
use the selected 9 features in d)i, the predicted class for training
set when l = 16is:
[0. 0. 0. ... 0. 0. 0.]
test accuracy when l = 16is:
0.865036231884058
use the selected 9 features in d)i, the predicted class for training
set when l = 17is:
[0. 0. 0. ... 0. 0. 0.]
test accuracy when l = 17is:
0.8695652173913043
use the selected 9 features in d)i, the predicted class for training
set when l = 18is:
[0. 0. 0. ... 0. 0. 0.]
test accuracy when l = 18is:
0.8695652173913043
use the selected 9 features in d)i, the predicted class for training
set when l = 19is:
[0. 0. 0. ... 0. 0. 0.]
test accuracy when l = 19is:
0.8680396643783371
use the selected 9 features in d)i, the predicted class for training
set when l = 20is:
[0. 0. 0. ... 0. 0. 0.]
test accuracy when l = 20is:
0.863768115942029
```

In [42]:

```
print("Using the selected 9 features, the accuracy is max when l = "+str(pse_acc
_9 .index(max(pse_acc_9 ))+1))
print("which is " +str(max(pse_acc_9 )))
```

```
Using the selected 9 features, the accuracy is max when l = 1
which is 0.8840579710144928
```

In [44]:

```
from sklearn.model_selection import StratifiedKFold
from sklearn.metrics import f1_score
```

In [601]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [602]:

```
#assuem select 10 features, use cv to determine the best l with best 10 features
score_report = list()
for l in range(20):
    l=l+1
    train_X =  createVar['df_bi_'+str(l)].iloc[:,0:-1]
    train_Y =  createVar['df_bi_'+str(l)]['label']
    model=LogisticRegression()
    cv=StratifiedKFold(n_splits=5,shuffle=False)
    scorelist = list()
    k=0
    for train_id, cv_id in cv.split(train_X,train_Y):
        k=k+1
        createVar['train_id_'+str(l)+'_'+str(k)]=train_id
        x_train_k, X_cv = train_X.loc[train_id], train_X.loc[cv_id]
        y_train_k, y_cv = train_Y.loc[train_id], train_Y.loc[cv_id]
        rfe=RFE(model,10)
        rfe=rfe.fit(x_train_k,y_train_k)
        createVar['sf_list_'+str(l)+'_'+str(k)]=list()
        for i in range(len(rfe.support_)):
            if rfe.support_[i]==True:
                createVar['sf_list_'+str(l)+'_'+str(k)].append(i)
        y_predict=model.fit(x_train_k[x_train_k.columns[ createVar['sf_list_'+st
r(l)+'_'+str(k)]]],
                              y_train_k).predict(X_cv[X_cv.columns[ createVar['sf_
list_'+str(l)+'_'+str(k)]]])
        scorelist.append(f1_score(y_cv, y_predict,average = 'weighted'))
    #print(l,scorelist)
    score_report.append(scorelist)
```

In [603]:

```
idx=0
mx=0
for i in range(20):
    a = np.mean(score_report[i])
    if mx < a :
        mx=a
        idx=i
    else:
        continue
```

In [604]:

```
idx2=0
mx=0
for i in range(5):
    a=score_report[idx][i]
    if mx < a :
        mx=a
        idx2=i
    else:
        continue
```

In [606]:

```
print("by using 5-folds, the accuracy reach max when l ="+str(idx+1))
```

```
by using 5-folds, the accuracy reach max when l =1
```

In [639]:

```
score_report[0]
```

Out[639]:

```
[0.9180952380952382,
 0.6938775510204083,
 0.8571428571428571,
 0.9180952380952382,
 0.8776223776223776]
```

In [608]:

```
for i in range(5):
    i=i+1
    print(createVar['signif_list_'+str(1)+'_'+str(i)])
```

```
[6, 8, 14, 15, 18, 20, 29, 32, 33, 34]
[0, 3, 6, 8, 14, 15, 19, 20, 29, 33]
[6, 8, 14, 19, 20, 22, 28, 29, 31, 33]
[6, 14, 15, 18, 19, 20, 28, 29, 33, 36]
[1, 6, 8, 14, 15, 19, 20, 28, 29, 33]
```

In [609]:

```
idx2=idx2+1
idx=idx+1
```

In [610]:

```
best_features=createVar['signif_list_'+str(idx)+'_'+str(idx2)]
```

In [611]:

```
best_features
```

Out[611]:

```
[6, 8, 14, 15, 18, 20, 29, 32, 33, 34]
```

In [54]:

```
col=['min1','max1','mean1','median1','std1','one_quartile1','th_quartile1',
                                                          'min2','max2','mean2','medi
an2','std2','one_quartile2','th_quartile2',
                                                          'min3','max3','mean3','medi
an3','std3','one_quartile3','th_quartile3',
                                                          'min4','max4','mean4','medi
an4','std4','one_quartile4','th_quartile4',
                                                          'min5','max5','mean5','medi
an5','std5','one_quartile5','th_quartile5',
                                                          'min6','max6','mean6','medi
an6','std6','one_quartile6','th_quartile6',
                                          ]
```

In [614]:

```
print("the best features are:")
for i in range(10):
    print(col[best_features[i]])
```

```
the best features are:
th_quartile1
max2
min3
max3
std3
th_quartile3
max5
std5
one_quartile5
th_quartile5
```

In [619]:

```
# extract best_features from the best 1

df_10 = dataset_1[:,best_features]
df_new_dataset_10 = np.column_stack((df_10 ,y_1))
df_b_10 = pd.DataFrame(df_new_dataset_10,columns = ['th_quartile1','max2','min3'
,'max3','std3','th_quartile3',
                                                     'max5','std5','one_quartile5
','th_quartile5','label'])

acc_10 = list()

clf = LogisticRegression(random_state = 0)
clf.fit(df_10,y_1.ravel())
y_pred_10= clf.predict(df_10)
acc_10 =accuracy_score(y_1,y_pred_10)
```

In [620]:

```
print("the accuracy of the moldel when train acuuracy is maximum is: "+str(acc_1
0))
```

```
the accuracy of the moldel when train acuuracy is maximum is: 0.9420
289855072463
```

In [621]:

```
print("the predicted classification is:")
print(y_pred_10)
```

```
the predicted classification is:
[1. 0. 1. 0. 0. 1. 1. 1. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

In [622]:

```
print("Apparently, the accuracy calculated by the right way is bigger than which
calculated by the wrong way. Thus we can't determin how many features and what f
eatures before using cross validation to determin the best l, we should use cros
s validation to find the best features.")
```

```
Apparently, the accuracy calculated by the right way is bigger than
which calculated by the wrong way. Thus we can't determin how many f
eatures and what features before using cross validation to determin
the best l, we should use cross validation to find the best features
.
```

In [60]:

```
#d)iv every testdata need a empty array to store 42-feartures, n pieces n*42
for i in range(20):
    i=i+1
    createVar['dataset2_'+str(i)]=np.empty(shape=[0, 42])
```

In [61]:

```
#split every test data(from 1 to 19) into n pieces, save into q1 to q19, each qi
have n dataframs. q are tuples
def finaldata2(n):
    for i in range(num_test):
        i=i+1
        createVar['q'+str(i)]= splitdata(createVar['testdata'+str(i)],n)


    for i in range(num_test):
        i=i+1
        for j in range(n):
            j=j+1
            createVar['dataset2_'+str(n)] = np.concatenate((createVar['dataset2_
'+str(n)],
                                                             give7features(create
Var['q'+str(i)][j-1])),axis=0)
```

In [62]:

```
for i in range(20):
    i=i+1
    finaldata2(i)
```

In [625]:

```
t_1=  np.zeros(shape=(1,19))
for i in range(0,4):
    t_1[0,i]=1
t_1 = t_1.T
```

In [624]:

```
from sklearn.metrics import confusion_matrix
print("the confusion matrix for train data:")
confusion_matrix(y_1, y_pred_10)
```

```
the confusion matrix for train data:
```

Out[624]:

```
array([[59,  1],
       [ 3,  6]])
```

In [627]:

```
print("the confusion matrix for test data:")
confusion_matrix(t_1, clf.fit(df_10,y_1.ravel()).predict(dataset2_1[:,best_featu
res]))
```

```
the confusion matrix for test data:
```

Out[627]:

```
array([[15,  0],
       [ 0,  4]])
```

In [628]:

```
from ggplot import *
from sklearn import metrics
```

In [629]:

```
probs = clf.predict_proba(df_10)
preds = probs[:,1]
fpr, tpr, threshold = metrics.roc_curve(y_1, preds)
roc_auc = metrics.auc(fpr, tpr)
```

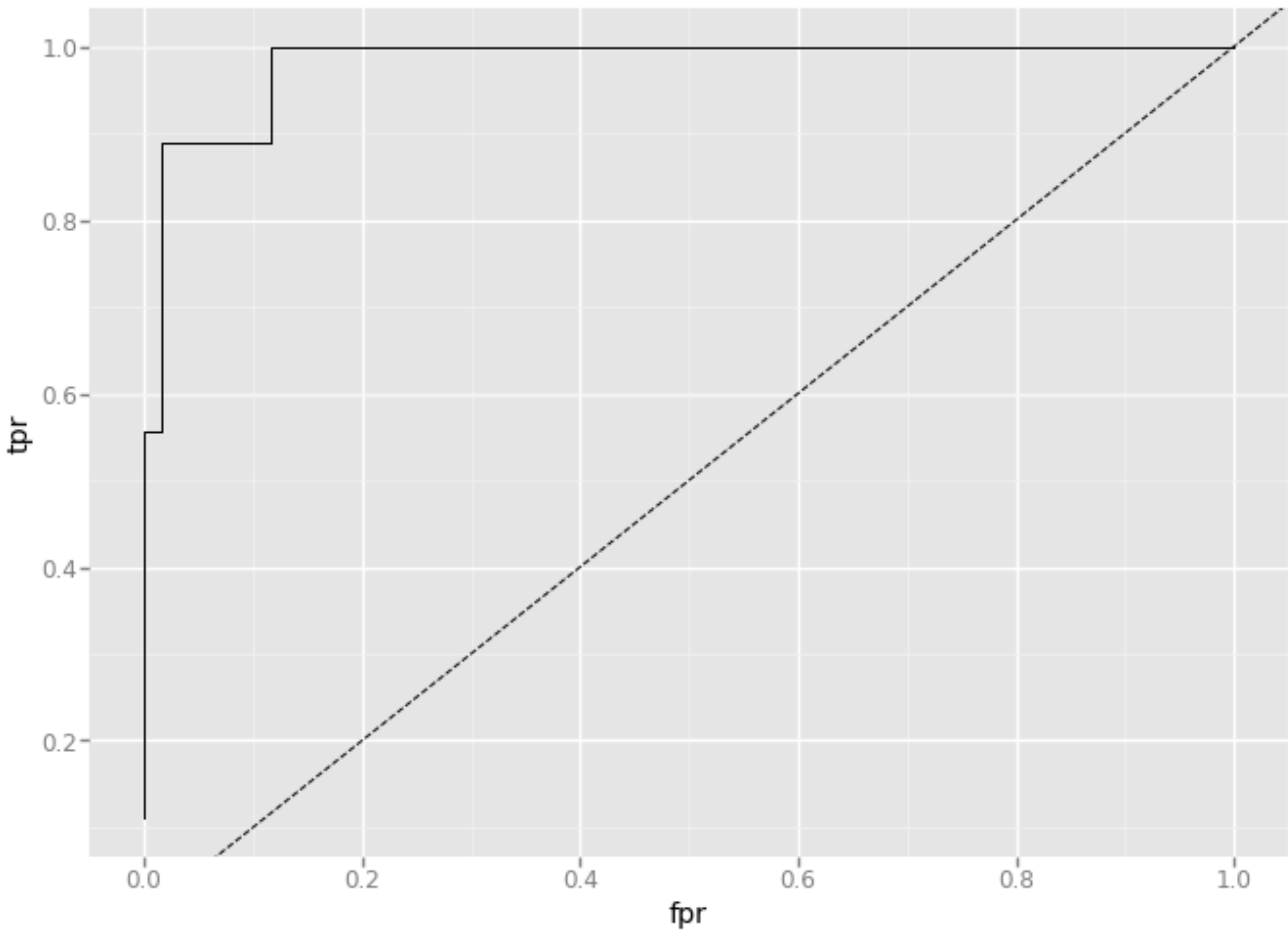
In [630]:

```
df = pd.DataFrame(dict(fpr = fpr, tpr = tpr))
```

In [635]:

```
print("ROC for train data:")
ggplot(df, aes(x = 'fpr', y = 'tpr')) + geom_line() + geom_abline(linetype = 'da
shed')
```

ROC for train data:



Out[635]:

```
<ggplot: (7024630571)>
```

In [636]:

```
from sklearn.metrics import roc_auc_score
auc=roc_auc_score(y_1, clf.decision_function(df_10))
print("auc for train data is "+str(auc))
```

auc for train data is 0.9814814814814814

In [70]:

```
df_2_dataset_10 = np.column_stack((df_10 ,y_1))
df_b2_10 = pd.DataFrame(df_2_dataset_10,columns = ['th_quartile1','max2','min3',
'max3','std3',
                                                   'th_quartile3','max5','std5',
'one_quartile5','th_quartile5','label'])
```

In [71]:

```
print("Refit with pruned set of features:")

import statsmodels.api as sm
y=df_b_10['label']
x=pd.DataFrame(df_b_10,columns=['th_quartile1','max2','min3','max3','std3','th_q
uartile3',
                                'max5','std5','one_quartile5','th_quartile5'])

logit_model=sm.Logit(y,x)
result=logit_model.fit()
print("summary",result.summary2())
```

```
Refit with pruned set of features:
Optimization terminated successfully.
         Current function value: 0.097756
         Iterations 13
summary                                Results: Logit
==============================================================
Model:              Logit            No. Iterations:   13.0000
Dependent Variable: label            Pseudo R-squared: 0.748
Date:               2018-07-01 12:56 AIC:              33.4903
No. Observations:   69               BIC:              55.8314
Df Model:           9                Log-Likelihood:   -6.7452
Df Residuals:       59               LL-Null:          -26.718
Converged:          1.0000           Scale:            1.0000
--------------------------------------------------------------
                 Coef.  Std.Err.    z    P>|z|   [0.025 0.975]
--------------------------------------------------------------
th_quartile1    -0.9523   0.6642 -1.4337 0.1516 -2.2541 0.3495
max2            -1.2553   1.1157 -1.1252 0.2605 -3.4420 0.9313
min3            -0.6225   0.4145 -1.5017 0.1332 -1.4350 0.1900
max3             1.7732   1.2262  1.4461 0.1482 -0.6301 4.1766
std3            -0.2923   1.3474 -0.2169 0.8283 -2.9332 2.3486
th_quartile3    -2.5766   1.9900 -1.2948 0.1954 -6.4769 1.3237
max5             0.9950   0.5591  1.7795 0.0752 -0.1009 2.0909
std5             0.9397   4.5550  0.2063 0.8365 -7.9878 9.8673
one_quartile5    0.7845   2.3130  0.3391 0.7345 -3.7490 5.3180
th_quartile5    -0.1483   1.8827 -0.0787 0.9372 -3.8383 3.5418
==============================================================
```

In [637]:

```
#d)v
x_test = dataset2_1[:,best_features]
clf.fit(df_10,y_1.ravel())
y_pr=clf.predict(x_test)
test_acc=accuracy_score(t_1,y_pr)
print("the predicted result in test set are " ,y_pr)
```

```
the predicted result in test set are  [1. 1. 1. 1. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

In [641]:

```
print("test accuracy is: ",test_acc)
print("test accuracy is bigger than the max cross validation accuracy which is "
+str(max(score_report[0])))
```

```
test accuracy is:  1.0
test accuracy is bigger than the max cross validation accuracy which
is 0.9180952380952382
```

In [74]:

```
#d)vi
print("if i choose 10 features, no well-separated occured")
```

```
if i choose 10 features, no well-separated occured
```

In [643]:

```
#d)vii
print("the confusion matrix from train data:")
print(confusion_matrix(y_1, y_pred_10))
print("it's imbalanced due to the class'0' is even more than 5 times of class'1'
")
```

```
the confusion matrix from train data:
[[59  1]
 [ 3  6]]
it's imbalanced due to the class'0' is even more than 5 times of cla
ss'1'
```

In [644]:

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import recall_score
from imblearn.over_sampling import SMOTE
```

In [645]:

```
xt, xv, yt,yv =train_test_split(df_10,y_1,test_size =.1,random_state=12)
```

In [646]:

```
sm=SMOTE(random_state=12,ratio=1.0)
xt_r,yt_r=sm.fit_sample(xt,yt.ravel())
```

In [647]:

```
log2=LogisticRegression()
log2.fit(xt_r,yt_r)
```

Out[647]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_interce
pt=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jo
bs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0
.0001,
          verbose=0, warm_start=False)
```

In [648]:

```
print("test accuracy when applying SMOTE is: ",accuracy_score(y_1,log2.predict(d
f_10)))
```

```
test accuracy when applying SMOTE is:  0.9565217391304348
```

In [649]:

```
print("the confusion matrix from train data after SMOTE:")
confusion_matrix(y_1, log2.predict(df_10))
```

```
the confusion matrix from train data after SMOTE:
```

Out[649]:

```
array([[57,  3],
       [ 0,  9]])
```

In [650]:

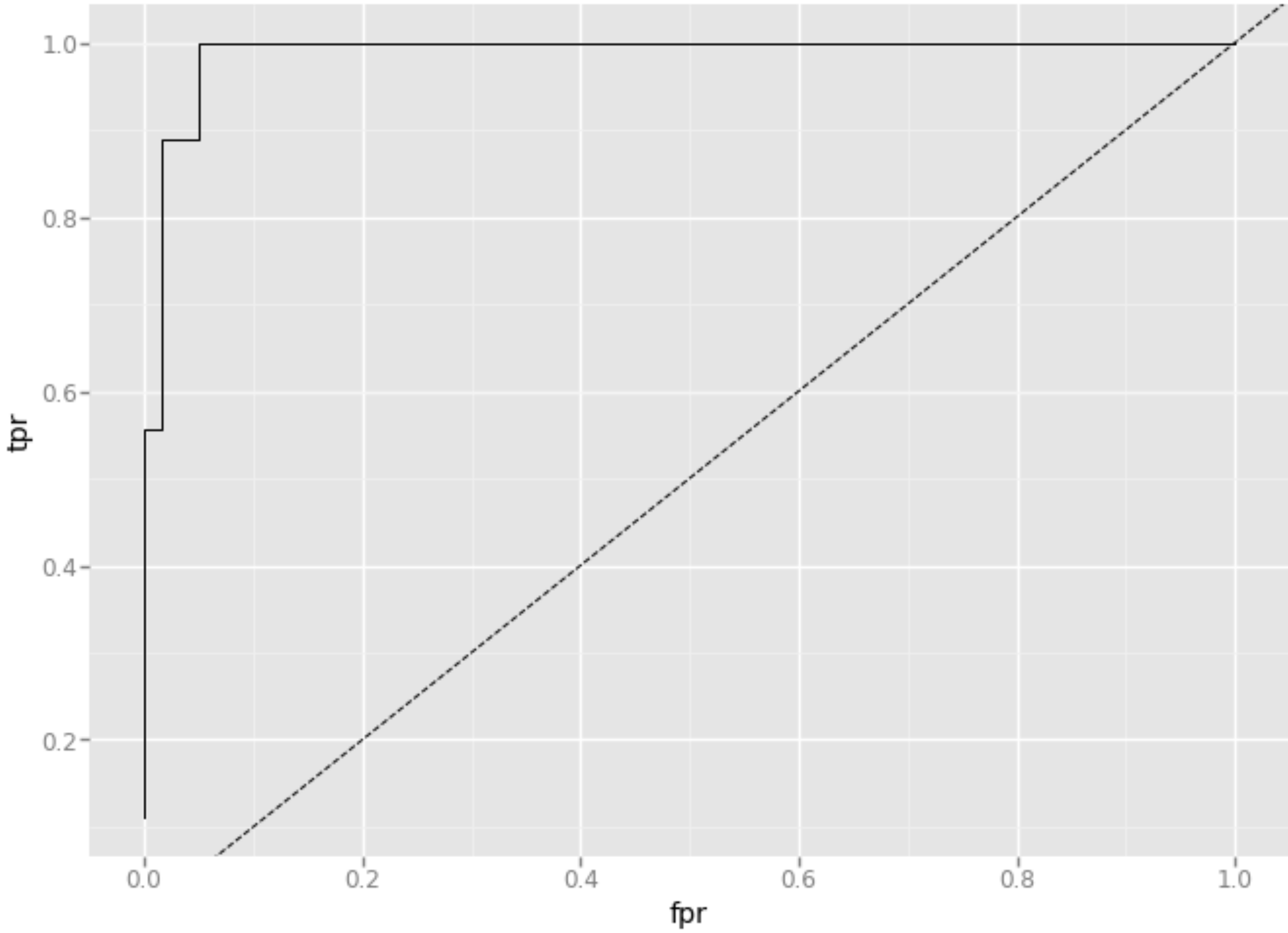
```
log2.predict(dataset2_1[:,best_features])
```

Out[650]:

```
array([1., 1., 1., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0
., 0.,
       0., 0.])
```

In [651]:

```
probs = log2.predict_proba(dataset_1[:,best_features])
preds = probs[:,1]
fpr, tpr, threshold = metrics.roc_curve(y_1, preds)
roc_auc = metrics.auc(fpr, tpr)
df = pd.DataFrame(dict(fpr = fpr, tpr = tpr))
ggplot(df, aes(x = 'fpr', y = 'tpr')) + geom_line() + geom_abline(linetype = 'da
shed')
```



Out[651]:

<ggplot: (7565875895)>

In [653]:

```
auc2=roc_auc_score(y_1, log2.decision_function(df_10))
print("auc after SMOTE is "+str(auc2))
```

auc after SMOTE is 0.9888888888888888

In [95]:

```
#e)i
from sklearn import preprocessing
import numpy as np
for i in range(20):
    i=i+1
    createVar['dataset_norm_'+str(i)] = preprocessing.scale(createVar['dataset_'
+str(i)])
```

In [160]:

```
for i in range(20):
    i=i+1
    createVar['dataset2_norm_'+str(i)] = preprocessing.scale(createVar['dataset2
_'+str(i)])
```

In [189]:

```
for i in range(20):
    i=i+1
    createVar['t_'+str(i)]=  np.zeros(shape=(1,19*i))
for i in range(20):
    i=i+1
    for j in range(0,4*i):
        createVar['t_'+str(i)][0,j]=1
for i in range(20):
    i=i+1
    createVar['t_'+str(i)] = createVar['t_'+str(i)].T
```

In [654]:

```
from sklearn.cross_validation import train_test_split
from sklearn.cross_validation import train_test_split,KFold
from sklearn.linear_model import LinearRegression, Lasso, Ridge
```

In [289]:

```
from sklearn.linear_model import LogisticRegressionCV
from sklearn.model_selection import cross_validate
from sklearn.cross_validation import cross_val_score
from sklearn import model_selection
```

In [655]:

```
score_report = list()
for l in range(20):
    l=l+1
    train_X =  createVar['dataset_norm_'+str(l)]
    train_Y =  createVar['y_'+str(l)]
    model=LogisticRegressionCV(penalty='l1', solver='liblinear',max_iter=200)
    scores = model_selection.cross_val_score(model, train_X,train_Y.ravel(), cv=
5)
    score_report.append(np.mean(scores))
```

In [657]:

```
score_report
```

Out[657]:

```
[0.8703296703296702,
 0.8341269841269842,
 0.8696864111498257,
 0.855064935064935,
 0.8376811594202899,
 0.8382603585071996,
 0.838659793814433,
 0.8514004914004915,
 0.8340903225806452,
 0.8565217391304347,
 0.8550975949808295,
 0.8527053669222344,
 0.8561638733705772,
 0.8509160835425458,
 0.8492753623188406,
 0.852373508844097,
 0.855100927441353,
 0.8478138359891177,
 0.8504832670594723,
 0.8550724637681159]
```

In [658]:

```
print("Using L1-penalized logistic regression, the best l is: ",score_report.ind
ex(max(score_report))+1)
```

```
Using L1-penalized logistic regression, the best l is:  1
```

In [659]:

```
train_X =  createVar['dataset_'+str(score_report.index(max(score_report))+1)]
train_Y =  createVar['y_'+str(score_report.index(max(score_report))+1)]
model=LogisticRegressionCV(penalty='l1', solver='liblinear')
y_pred=model.fit(train_X ,train_Y.ravel()).predict(createVar['dataset2_'+
                                                              str(score_report.in
dex(max(score_report))+1)])
print("The accuracy when use L1 penalty to classify 2 classes is :",
      accuracy_score( createVar['t_'+str(score_report.index(max(score_report))+1
)], y_pred))
```

```
The accuracy when use L1 penalty to classify 2 classes is : 1.0
```

In [660]:

```
print("Compared to p-value selection, L1 perform better. Because in previous met
hod, it is almost impossible to do cross validation for all possible combination
s of different amount of features, however, using L1, it can somehow check all p
ossibilities and give the same result as p-value do.")
```

```
Compared to p-value selection, L1 perform better. Because in previou
s method, it is almost impossible to do cross validation for all pos
sible combinations of different amount of features, however, using L
1, it can somehow check all possibilities and give the same result a
s p-value do.
```

In [370]:

```
for i in range(69):
    i=i+1
    createVar['ym_'+str(i)]=  np.zeros(shape=(1,69*i))
for i in range(20):
    i=i+1
    for j in range(0,5*i):
        createVar['ym_'+str(i)][0,j]=1
    for j in range(5*i,9*i):
        createVar['ym_'+str(i)][0,j]=2
    for j in range(9*i,21*i):
        createVar['ym_'+str(i)][0,j]=3
    for j in range(21*i,33*i):
        createVar['ym_'+str(i)][0,j]=4
    for j in range(33*i,45*i):
        createVar['ym_'+str(i)][0,j]=5
    for j in range(45*i,57*i):
        createVar['ym_'+str(i)][0,j]=6
    for j in range(57*i,69*i):
        createVar['ym_'+str(i)][0,j]=7
for i in range(20):
    i=i+1
    createVar['ym_'+str(i)] = createVar['ym_'+str(i)].T
```

In [371]:

```
for i in range(20):
    i=i+1
    createVar['tm_'+str(i)]=  np.zeros(shape=(1,19*i))
for i in range(20):
    i=i+1
    for j in range(0,2*i):
        createVar['tm_'+str(i)][0,j]=1
    for j in range(2*i,4*i):
        createVar['tm_'+str(i)][0,j]=2
    for j in range(4*i,7*i):
        createVar['tm_'+str(i)][0,j]=3
    for j in range(7*i,10*i):
        createVar['tm_'+str(i)][0,j]=4
    for j in range(10*i,13*i):
        createVar['tm_'+str(i)][0,j]=5
    for j in range(13*i,16*i):
        createVar['tm_'+str(i)][0,j]=6
    for j in range(16*i,19*i):
        createVar['tm_'+str(i)][0,j]=7
for i in range(20):
    i=i+1
    createVar['tm_'+str(i)] = createVar['tm_'+str(i)].T
```

In [530]:

```
score_report2 = list()
for l in range(20):
    l=l+1
    train_X =  createVar['dataset_norm_'+str(l)]
    train_Y =  createVar['ym_'+str(l)]
    model2=LogisticRegressionCV(penalty='l1', solver='saga',multi_class='multino
mial')
    scores = model_selection.cross_val_score(model, train_X,train_Y.ravel(), cv=
4)
    score_report2.append(np.mean(scores))
    print (l)
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

In [661]:

```
score_report2
```

Out[661]:

```
[0.5081699346405228,
 0.49264705882352944,
 0.4686085972850679,
 0.4891304347826087,
 0.5333132852178563,
 0.475658140403286,
 0.4223829201101928,
 0.5072463768115941,
 0.47336641852770883,
 0.4825077295335394,
 0.4545182400445559,
 0.4070048309178744,
 0.40251984126984125,
 0.4202573642879188,
 0.4280177187153932,
 0.4003623188405797,
 0.4058415174943698,
 0.4266958821698994,
 0.4698198702170508,
 0.40072463768115946]
```

In [662]:

```
print("Using L1-penalized logistic regression to classifiy 6 classes, the best l
is: ",score_report2.index(max(score_report2))+1)
```

```
Using L1-penalized logistic regression to classifiy 6 classes, the b
est l is:  5
```

In [683]:

```
train_X =  createVar['dataset_'+str(score_report2.index(max(score_report2))+1)]
train_Y =  createVar['ym_'+str(score_report2.index(max(score_report2))+1)]

model2=LogisticRegressionCV(penalty='l1', solver='saga',multi_class='multinomial
')

y_true= createVar['tm_'+str(score_report2.index(max(score_report2))+1)]
y_pred=model2.fit(train_X ,train_Y.ravel()).predict(createVar['dataset2_'+str(sc
ore_report2.index(max(score_report2))+1)])
y_score=model2.predict_proba(createVar['dataset2_'+str(score_report2.index(max(s
core_report2))+1)])

print("The accuracy when use L1 penalty to classify 6 classes is :",accuracy_sco
re(y_true, y_pred))

from sklearn.metrics import classification_report
print ("classification_report(left: labels):")
print (confusion_matrix(y_true,y_pred))
```

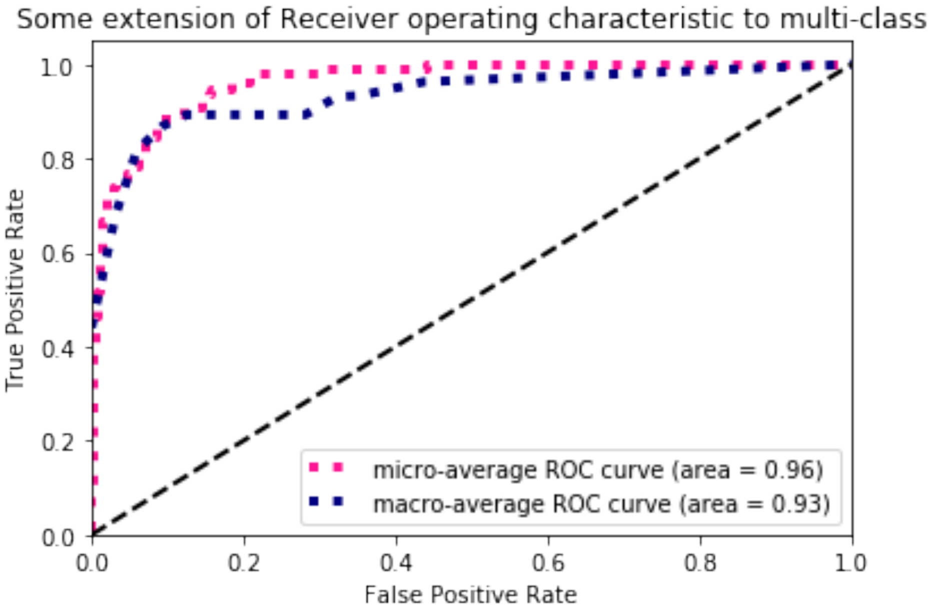
```
The accuracy when use L1 penalty to classify 6 classes is : 0.747368
4210526316
classification_report(left: labels):
[[ 6  1  3  0  0  0  0]
 [ 1  3  2  0  3  0  1]
 [ 0  0 15  0  0  0  0]
 [ 0  0  0 12  1  2  0]
 [ 0  0  0  2 11  2  0]
 [ 0  0  0  2  4  9  0]
 [ 0  0  0  0  0  0 15]]
```

In [664]:

```
n_classes=7
from scipy import interp
from sklearn.metrics import roc_curve, auc
```

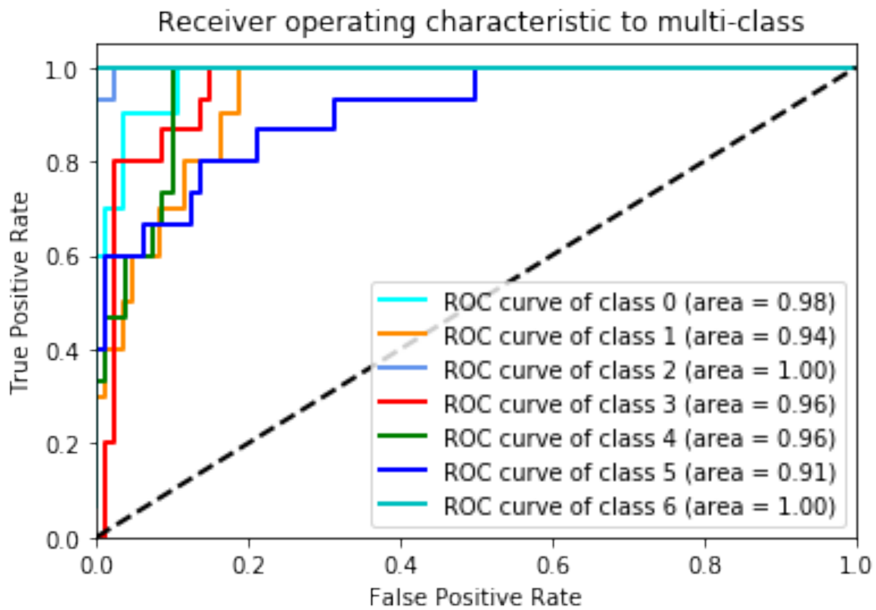
In [701]:

```
from sklearn.preprocessing import label_binarize
y = label_binarize(y_true, classes=[1,2,3,4,5,6,7])
fpr["micro"], tpr["micro"], _ = roc_curve(y.ravel(), y_score.ravel())
roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])

all_fpr = np.unique(np.concatenate([fpr[i] for i in range(n_classes)]))

# Then interpolate all ROC curves at this points
mean_tpr = np.zeros_like(all_fpr)
for i in range(n_classes):
    mean_tpr += interp(all_fpr, fpr[i], tpr[i])

# Finally average it and compute AUC
mean_tpr /= n_classes

fpr["macro"] = all_fpr
tpr["macro"] = mean_tpr
roc_auc["macro"] = auc(fpr["macro"], tpr["macro"])

# Plot all ROC curves
plt.figure()
plt.plot(fpr["micro"], tpr["micro"],
         label='micro-average ROC curve (area = {0:0.2f})'
               ''.format(roc_auc["micro"]),
         color='deeppink', linestyle=':', linewidth=4)

plt.plot(fpr["macro"], tpr["macro"],
         label='macro-average ROC curve (area = {0:0.2f})'
               ''.format(roc_auc["macro"]),
         color='navy', linestyle=':', linewidth=4)
plt.plot([0, 1], [0, 1], 'k--', lw=lw)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Some extension of Receiver operating characteristic to multi-class')
plt.legend(loc="lower right")
plt.show()
```


Some extension of Receiver operating characteristic to multi-class
True Positive Rate
False Positive Rate
micro-average ROC curve (area = 0.96)
macro-average ROC curve (area = 0.93)
0.0
0.2
0.4
0.6
0.8
1.0

In [685]:

```
from sklearn.metrics import roc_curve, auc
from itertools import cycle
import matplotlib.pyplot as plt
fpr = dict()
tpr = dict()
roc_auc = dict()
for i in range(7):
    fpr[i], tpr[i], _ = roc_curve(y[:, i], y_score[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

lw=2
colors = cycle(['aqua', 'darkorange', 'cornflowerblue','r','g','b','c'])
for i, color in zip(range(7), colors):
    plt.plot(fpr[i], tpr[i], color=color, lw=lw,
             label='ROC curve of class {0} (area = {1:0.2f})'
             ''.format(i, roc_auc[i]))

plt.plot([0, 1], [0, 1], 'k--', lw=lw)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title(' Receiver operating characteristic to multi-class')
plt.legend(loc="lower right")
plt.show()
```



In [686]:

```
score_report3 = list()
from sklearn.naive_bayes import GaussianNB
for l in range(20):
    l=l+1
    train_X =  createVar['dataset_norm_'+str(l)]
    train_Y =  createVar['ym_'+str(l)]
    gnb = GaussianNB()
    scores = model_selection.cross_val_score(gnb , train_X,train_Y.ravel(), cv=4
)
    score_report3.append(np.mean(scores))
```

In [687]:

```
score_report3
```

Out[687]:

```
[0.6086601307189543,
 0.5802521008403362,
 0.546285822021162,
 0.5652173913043479,
 0.585304731355252,
 0.5553584764749813,
 0.5630337465564739,
 0.5489130434782608,
 0.5153122415219189,
 0.5200631805350182,
 0.5559245335561125,
 0.5289855072463768,
 0.5227777777777778,
 0.4999142690579884,
 0.526499506150669,
 0.5181159420289855,
 0.4969439994427805,
 0.51202676070947,
 0.5369280040277468,
 0.5130434782608696]
```

In [688]:

```
print("Using Gausian Naive Bayes to classifiy 6 classes, the best l is: ",score_
report3.index(max(score_report3))+1)
```

```
Using Gausian Naive Bayes to classifiy 6 classes, the best l is:  1
```

In [689]:

```
train_X =  createVar['dataset_'+str(score_report3.index(max(score_report3))+1)]
train_Y =  createVar['ym_'+str(score_report3.index(max(score_report3))+1)]
gnb = GaussianNB()
y_true3=createVar['tm_'+str(score_report3.index(max(score_report3))+1)]
y_pred3=gnb.fit(train_X ,train_Y.ravel()).predict(createVar['dataset2_'+str(scor
e_report3.index(max(score_report3))+1)])

y_score3=gnb.predict_proba(createVar['dataset2_'+str(score_report3.index(max(sco
re_report3))+1)])

print("The accuracy when use Gausian Naive Bayes'to classify 6 classes is :",acc
uracy_score(y_true3, y_pred3))
```

```
The accuracy when use Gausian Naive Bayes'to classify 6 classes is :
0.8947368421052632
```

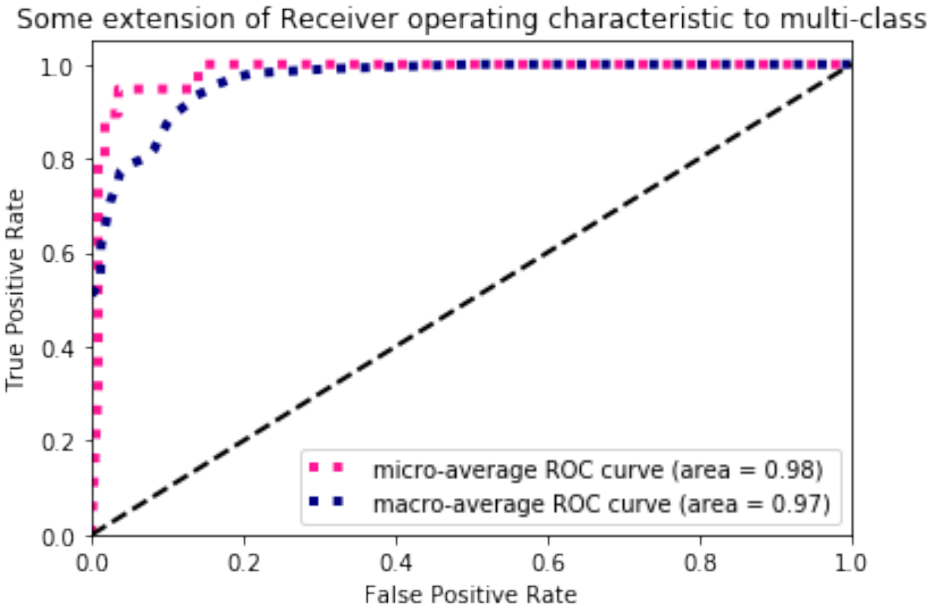
In [690]:

```
print ("classification_report(left: labels):")
print (confusion_matrix(y_true3,y_pred3))
```

```
classification_report(left: labels):
[[2 0 0 0 0 0 0]
 [0 1 0 1 0 0 0]
 [0 0 3 0 0 0 0]
 [0 0 0 3 0 0 0]
 [0 0 0 0 2 1 0]
 [0 0 0 0 0 3 0]
 [0 0 0 0 0 0 3]]
```

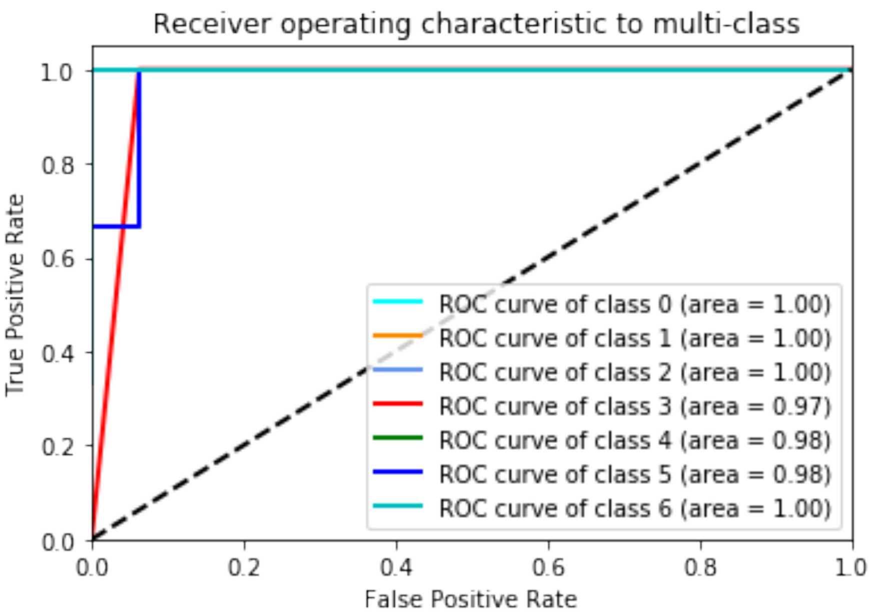
In [691]:

```
from sklearn.preprocessing import label_binarize
y3 = label_binarize(y_true3, classes=[1,2,3,4,5,6,7])
fpr["micro"], tpr["micro"], _ = roc_curve(y3.ravel(), y_score3.ravel())
roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])

all_fpr = np.unique(np.concatenate([fpr[i] for i in range(n_classes)]))

# Then interpolate all ROC curves at this points
mean_tpr = np.zeros_like(all_fpr)
for i in range(n_classes):
    mean_tpr += interp(all_fpr, fpr[i], tpr[i])

# Finally average it and compute AUC
mean_tpr /= n_classes

fpr["macro"] = all_fpr
tpr["macro"] = mean_tpr
roc_auc["macro"] = auc(fpr["macro"], tpr["macro"])

# Plot all ROC curves
plt.figure()
plt.plot(fpr["micro"], tpr["micro"],
         label='micro-average ROC curve (area = {0:0.2f})'
               ''.format(roc_auc["micro"]),
         color='deeppink', linestyle=':', linewidth=4)

plt.plot(fpr["macro"], tpr["macro"],
         label='macro-average ROC curve (area = {0:0.2f})'
               ''.format(roc_auc["macro"]),
         color='navy', linestyle=':', linewidth=4)
plt.plot([0, 1], [0, 1], 'k--', lw=lw)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Some extension of Receiver operating characteristic to multi-class')
plt.legend(loc="lower right")
plt.show()
```


Some extension of Receiver operating characteristic to multi-class
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
micro-average ROC curve (area = 0.98)
macro-average ROC curve (area = 0.97)

In [692]:

```
from sklearn.preprocessing import label_binarize
y3 = label_binarize(y_true3, classes=[1,2,3,4,5,6,7])


from sklearn.metrics import roc_curve, auc
from itertools import cycle
import matplotlib.pyplot as plt
fpr = dict()
tpr = dict()
roc_auc = dict()
for i in range(7):
    fpr[i], tpr[i], _ = roc_curve(y3[:, i], y_score3[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

lw=2
colors = cycle(['aqua', 'darkorange', 'cornflowerblue','r','g','b','c'])
for i, color in zip(range(7), colors):
    plt.plot(fpr[i], tpr[i], color=color, lw=lw,
             label='ROC curve of class {0} (area = {1:0.2f})'
             ''.format(i, roc_auc[i]))

plt.plot([0, 1], [0, 1], 'k--', lw=lw)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title(' Receiver operating characteristic to multi-class')
plt.legend(loc="lower right")
plt.show()
```



In [693]:

```
from sklearn.naive_bayes import MultinomialNB
score_report4 = list()
for l in range(20):
    l=l+1
    train_X =  createVar['dataset_'+str(l)]
    train_Y =  createVar['ym_'+str(l)]
    mnb = MultinomialNB()
    #y_pred = gnb.fit(createVar['dataset_norm_'+str(l)], createVar['ym_'+str(i)]
).predict(createVar['dataset_norm_'+str(l))
    scores = model_selection.cross_val_score(mnb , train_X,train_Y.ravel(), cv=5
)
    score_report4.append(np.mean(scores))
```

In [694]:

```
score_report4
```

Out[694]:

```
[0.4639928698752228,
 0.47431998286571,
 0.46007423117709445,
 0.47277444273814506,
 0.4608695652173913,
 0.4592728969279357,
 0.47116329274874447,
 0.4569510824237589,
 0.46276147299769355,
 0.4666666666666667,
 0.4691318467200693,
 0.45013696322090535,
 0.46306967820464884,
 0.4556291828472279,
 0.44734299516908216,
 0.457609689282273 1,
 0.45584753602282885,
 0.45605043255261013,
 0.46481895887556257,
 0.455072463768116]
```

In [695]:

```
print("Using  multinomial Naive Bayes to classifiy 6 classes, the best l is: ",s
core_report4.index(max(score_report4))+1)
```

```
Using  multinomial Naive Bayes to classifiy 6 classes, the best l is
:  2
```

In [696]:

```
train_X =  createVar['dataset_'+str(score_report4.index(max(score_report4))+1)]
train_Y =  createVar['ym_'+str(score_report4.index(max(score_report4))+1)]
mnb = MultinomialNB()
y_true4=createVar['tm_'+str(score_report4.index(max(score_report4))+1)]
y_pred4=mnb.fit(train_X ,train_Y.ravel()).predict(createVar['dataset2_'+str(scor
e_report4.index(max(score_report4))+1)])
y_score4=gnb.predict_proba(createVar['dataset2_'+str(score_report4.index(max(sco
re_report4))+1)])

print("The accuracy when use Gausian Naive Bayes multinomial to classify 6 class
es is :",accuracy_score(y_true4, y_pred4))
```

```
The accuracy when use Gausian Naive Bayes multinomial to classify 6
classes is : 0.7105263157894737
```

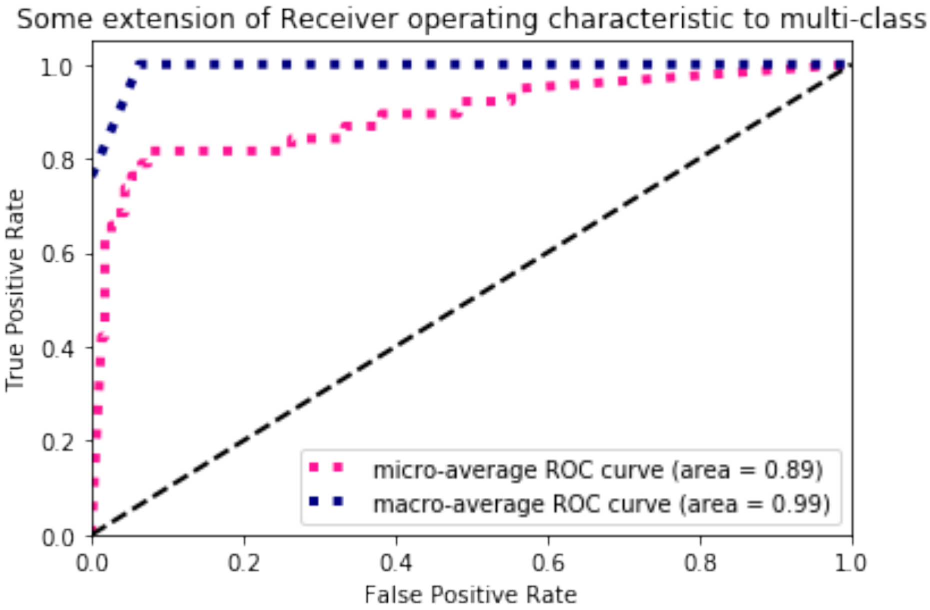
In [697]:

```
print ("classification_report(left: labels):")
print (confusion_matrix(y_true4,y_pred4))
```

```
classification_report(left: labels):
[[4 0 0 0 0 0 0]
 [2 1 1 0 0 0 0]
 [0 0 5 0 0 0 1]
 [0 0 0 2 1 3 0]
 [0 0 0 1 4 1 0]
 [0 0 0 1 0 5 0]
 [0 0 0 0 0 0 6]]
```

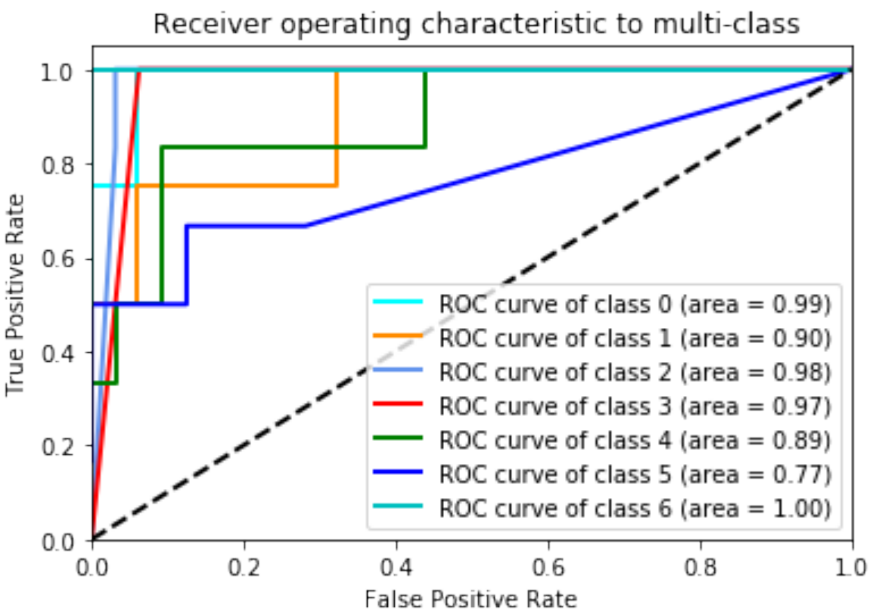
In [698]:

```
from sklearn.preprocessing import label_binarize
y4= label_binarize(y_true4, classes=[1,2,3,4,5,6,7])
fpr["micro"], tpr["micro"], _ = roc_curve(y4.ravel(), y_score4.ravel())
roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])

all_fpr = np.unique(np.concatenate([fpr[i] for i in range(n_classes)]))

# Then interpolate all ROC curves at this points
mean_tpr = np.zeros_like(all_fpr)
for i in range(n_classes):
    mean_tpr += interp(all_fpr, fpr[i], tpr[i])

# Finally average it and compute AUC
mean_tpr /= n_classes

fpr["macro"] = all_fpr
tpr["macro"] = mean_tpr
roc_auc["macro"] = auc(fpr["macro"], tpr["macro"])

# Plot all ROC curves
plt.figure()
plt.plot(fpr["micro"], tpr["micro"],
         label='micro-average ROC curve (area = {0:0.2f})'
               ''.format(roc_auc["micro"]),
         color='deeppink', linestyle=':', linewidth=4)

plt.plot(fpr["macro"], tpr["macro"],
         label='macro-average ROC curve (area = {0:0.2f})'
               ''.format(roc_auc["macro"]),
         color='navy', linestyle=':', linewidth=4)
plt.plot([0, 1], [0, 1], 'k--', lw=lw)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Some extension of Receiver operating characteristic to multi-class')
plt.legend(loc="lower right")
plt.show()
```


Some extension of Receiver operating characteristic to multi-class
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
micro-average ROC curve (area = 0.89)
macro-average ROC curve (area = 0.99)

In [699]:

```
from sklearn.preprocessing import label_binarize
y4 = label_binarize(y_true4, classes=[1,2,3,4,5,6,7])


from sklearn.metrics import roc_curve, auc
from itertools import cycle
import matplotlib.pyplot as plt
fpr = dict()
tpr = dict()
roc_auc = dict()
for i in range(7):
    fpr[i], tpr[i], _ = roc_curve(y4[:, i], y_score4[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

lw=2
colors = cycle(['aqua', 'darkorange', 'cornflowerblue','r','g','b','c'])
for i, color in zip(range(7), colors):
    plt.plot(fpr[i], tpr[i], color=color, lw=lw,
             label='ROC curve of class {0} (area = {1:0.2f})'
             ''.format(i, roc_auc[i]))

plt.plot([0, 1], [0, 1], 'k--', lw=lw)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title(' Receiver operating characteristic to multi-class')
plt.legend(loc="lower right")
plt.show()
```



In [704]:

```
print("Based on the data, using Gausian Naive Bayes is the best way in this prob
lem.")
print("because the test accuracy is the highest and also computational friendly"
)
```

```
Based on the data, using Gausian Naive Bayes is the best way in this
problem.
because the test accuracy is the highest and also computational frie
ndly
```