統計學習初論（105-2）

作業三 (Part A)

作業設計：盧信銘
國立台灣大學資管系

截止時間：2017 年 4 月 18 日上午 9 點

第一題請至 RSAND 上批改，範例命令為 (第 a 小題)：
sl_check_hw3q1a ./your_program，第 b 小題為：sl_check_hw3q1b ./your_program。
作業自己做。嚴禁抄襲。不接受紙本繳交，不接受遲交。請以英文或中文作答。

第一題

(60 points) We are going to explore the problem of identifying smartphone position through probabilistic generative models. Motion sensors in smartphones provide valuable information for researchers to understand its owners. An interesting (and more challenging) task is to identify human activities through the data recorded by motion sensors. For example, we want to know whether the smartphone owner is walking, running, or biking. In this homework problem, we are going to tackle a simpler problem. We want to know the static position of the smartphone. There are six possible positions:

- Phoneonback: The phone is laying on the back of the phone with the screen pointing up (away from the ground).
- Phoneonfront: The phone is laying on the back of the phone with the screen pointing towards the ground
- Phoneonbottom: The phone is standing on the bottom of the screen, meaning the bottom is pointed towards the ground
- Phoneontop: The phone is standing on the top of the screen, meaning the top is pointed towards the ground
- Phoneonleft: The phone is laying on the left side of the screen.
- Phoneonright: The phone is laying on the right side of the screen.

The input data is the reading of the accelerometer (cf. https://en.wikipedia.org/wiki/Accelerometer) in the smartphone. We have a training dataset that contains about 28,500 data points for phones in each of the six positions. The following is the first six data points for the first position (Phoneonback):

```
> head(traindata[[1]])
          x           y        z
1 0.1388092  0.074340820 9.801056
```

```
2 0.1649933  0.006500244 9.690369
3 0.2114105 -0.001831055 9.755829
4 0.1840363 -0.007781982 9.774872
5 0.1423798  0.005310059 9.765350
6 0.1852264  0.026733398 9.829620
```

We are going to train a model that can predict smartphone positions given its accelerometer readings. To achieve this goal, we are going to divide this question into two sub-questions. The first is about training a generative classifier, and the second is to predict smartphone positions.

(a) (30 points) To train probabilistic generative classifiers, we need to assume how the data are generated in one given position. Here we are going to adopt a simple assumption that the data are generated from multivariate Gaussian distributions. To train a model under this assumption, we need to compute the mean and variance of the readings from the training data in phonetrain.rdata. This data file contains two variables. The first variable is outclass, which is a vector of strings of possible phone positions:

```
> outclass
[1] "Phoneonback"   "Phoneonbottom" "Phoneonfront"  "Phoneonleft"   "Phoneonright"
[6] "Phoneontop"
```

The second variable is traindata, which is a list of length 6. Each component is a data frame that are training data for a particular position. The 6 components in traindata have the same order as the strings in outclass. That is, traindata[[1]] is for Phoneonback, traindata[[2]] is for Phoneonbottom, and so on. For the data frame in each position, compute the mean (named mu1), covariance (use unbiased estimator; named sigma1), precision (the inverse of covariance; named prec1), logarithm of determinant of sigma1 (named detsig_log), and number of observations in this data frame (named N1). An example command for the construction of this list data structure is:

```
list(mu1=mu1, sigma1=cov1, prec1=solve(cov1), detsig_log=logdet, N1=N1)
```

The trained model is a list of six components, ordered by the position strings in outclass. Each component is a list of parameters for a position as explained above.

Write a function named pgm_train for this task. This function takes two input parameters. The first is outclass, and the second is alldata, which is the list of six data frames for the training data of different positions. This function should return the trained model as explained above.

Input and output, sample 1

```
> setwd('your_path')
> load('phonetrain.rdata')
>
> train2 = list()
> for(aclass in outclass) {
+     train2[[aclass]] = traindata[[aclass]][1:500,]
+ }
> model1=pgm_train(outclass, train2)

> model1[1]
$Phoneonback
```

```
$Phoneonback$mu1
         x          y          z 
0.16199402 0.01477441 9.73784279 

$Phoneonback$sigma1
              x             y             z
x  6.382636e-04 -6.211061e-05  1.310078e-05
y -6.211061e-05  6.812823e-04 -3.367878e-05
z  1.310078e-05 -3.367878e-05  1.001629e-03

$Phoneonback$prec1
           x          y          z
x 1581.02638  143.35381  -15.85886
y  143.35381 1483.26227   47.99823
z  -15.85886   47.99823 1000.19502

$Phoneonback$detsig_log
  modulus 
-21.56515 

$Phoneonback$N1
[1] 500


> model1[3]
$Phoneonfront
$Phoneonfront$mu1
          x           y           z 
 0.08098285  0.31618890 -9.75056005 

$Phoneonfront$sigma1
              x            y             z
x  6.044951e-04 5.133011e-06 -6.190712e-05
y  5.133011e-06 6.353561e-04  1.661259e-06
z -6.190712e-05 1.661259e-06  1.022019e-03

$Phoneonfront$prec1
          x          y         z
x 1664.7188  -13.71290 100.85993
y  -13.7129 1574.04022  -3.38919
z  100.8599   -3.38919 984.57062

$Phoneonfront$detsig_log
  modulus 
-21.66472 

$Phoneonfront$N1
[1] 500


> model1[5]
$Phoneonright
$Phoneonright$mu1
          x           y           z 
-9.65647737  0.15743481 -0.07427301 

$Phoneonright$sigma1
```

```
              x             y             z
x  6.221814e-04 -1.944180e-05  1.720659e-05
y -1.944180e-05  6.040970e-04 -2.081885e-05
z  1.720659e-05 -2.081885e-05  1.090656e-03

$Phoneonright$prec1
           x          y          z
x 1609.51604   50.95788 -24.41962
y   50.95788 1658.06633  30.84586
z  -24.41962   30.84586 917.85355

$Phoneonright$detsig_log
 modulus
-21.6171

$Phoneonright$N1
[1] 500
```

Input and output, sample 2

```
> train3 = list()
> for(aclass in outclass) {
+     train3[[aclass]] = traindata[[aclass]][1:1500,]
+ }
> model2=pgm_train(outclass, train3)
> model2[2]
$Phoneonbottom
$Phoneonbottom$mu1
        x         y         z
0.1444420 9.8831614 0.1100169

$Phoneonbottom$sigma1
              x             y             z
x  6.624064e-04 -1.087213e-07 -4.537591e-06
y -1.087213e-07  6.420947e-04 -2.331090e-06
z -4.537591e-06 -2.331090e-06  1.004589e-03

$Phoneonbottom$prec1
             x            y          z
x 1509.6939263    0.2803843   6.819734
y    0.2803843 1557.4159359   3.615160
z    6.8197338    3.6151603 995.471607

$Phoneonbottom$detsig_log
  modulus
-21.57362

$Phoneonbottom$N1
[1] 1500


> model2[4]
$Phoneonleft
$Phoneonleft$mu1
          x            y            z
```

```
 9.89267651  0.27117529 -0.02911975

$Phoneonleft$sigma1
              x             y             z
x  6.720721e-04 -1.761831e-05 -9.695159e-07
y -1.761831e-05  6.633099e-04  3.235197e-05
z -9.695159e-07  3.235197e-05  1.040748e-03

$Phoneonleft$prec1
             x          y            z
x 1488.9724017   39.54120    0.1579119
y   39.5412017 1510.93011  -46.9308837
z    0.1579119  -46.93088  962.3061929

$Phoneonleft$detsig_log
  modulus
-21.49344

$Phoneonleft$N1
[1] 1500


> model2[6]
$Phoneontop
$Phoneontop$mu1
          x            y            z
 0.03439581 -9.54268689 -0.27299432

$Phoneontop$sigma1
              x             y             z
x  6.796703e-04  1.472893e-05 -4.098093e-06
y  1.472893e-05  6.685889e-04 -2.093863e-05
z -4.098093e-06 -2.093863e-05  1.013832e-03

$Phoneontop$prec1
           x          y         z
x 1472.03272  -32.26322   5.28389
y  -32.26322 1497.36257  30.79454
z    5.28389   30.79454 987.01366

$Phoneontop$detsig_log
  modulus
-21.49941

$Phoneontop$N1
[1] 1500
```

Evaluation: All credits will be given based on the correctness of 10 testing cases. Correct output in a case is worth 3 points.

(b) (30 points) To predict phone positions, compute the posterior of each position given the trained model, and select the position with the largest probability. That is, given the reading $g = \left(g_x, g_y, g_z\right)^T$, the probability that the phone position is k can be written as $Prob(POS = k|g) \propto Prob(g|POS = k)Prob(POS = k)$.

We do not prefer any position so a reasonable setting is $Prob(POS = k) = c$, for $k = 1, 2, \dots, 6$. Here c is a constant, and this constant plays no role in influencing the prediction outcome.

As for $Prob(g|POS = k)$, since we adopted Gaussian assumption in this problem, $Prob(g|POS = k) = N(g|\mu_k, \Sigma_k)$, where μ_k and Σ_k are learned from the training data. Let $q_k = Prob(g|POS = k)$, then the prediction of phone position can be determined by looking at $q = (q_1, q_2, q_3, q_4, q_5, q_6)$, and select the one with the largest value.

Write a function named pgm_predict, This function takes two parameters. The first parameter, amodel, is the data structure of the trained model. The second parameter, testdata, is a data frame or matrix that contains accelerometer readers to be processed. You should check testdata and make sure it contains three columns. Return NULL if this condition is not satisfied.

Predict the phone position for data points in each row. Return a vector that contain the predicted phone position corresponding to testdata. You should use 1, 2, 3, 4, 5, and 6 to represent predicted cell phone positions. These integers corresponds to the strings in outclass.

Sample input and output:

```
> load('phonetrain.rdata')
> load('phonetest1.rdata')
>
> model1=pgm_train(outclass, traindata)
> pred1=pgm_predict(model1, testds1_feature)
>
>
>
> pred1[1:50]
 [1] 3 1 1 3 2 3 6 2 6 4 3 1 5 2 1 3 4 2 2 4 4 3 1 3 5 5 2 4 6 5 4 3 2
5 2 5 3 3 5 4 4 3 1
[44] 3 1 3 5 3 5 5
```

Evaluation: All credits will be given based on the correctness of 10 testing cases. Correct output in a case is worth 3 points.