Homework 4 (Part 2)

Ying-Yu Huang (yingyu010365@gmail.com)

Question 1: Download and Describe your choice of a real data set DS

Step 1: Describe the original classification task for DS:

It's a Vicon motion capture camera system that used to record users performing 5 hand postures with markers attached to a left-handed glove.

The original data has 78096 cases, each cases has (x0, y0, z0) to (x10, y10, z10) features. The features indicate different local coordinate system for the hand, and we use these coordinates to determine the posture that users perform.

The original data has 5 classes, 1= Fist (with thumb out), 2=Stop (hand flat), 3=Point1(point with pointer finger), 4=Point2(point with pointer and middle fingers), 5=Grab (fingers curled as if to grab), this is the 5 different postures that the users perform. The features are continuous.

Step 2: Describe precisely the reduced data set RDS

I choose class 2, 3, 5 be my three classes. 2=Stop (hand flat), 3=Point1(point with pointer finger), 5=Grab (fingers curled as if to grab), because they have less missing values, and each classes has 2000 cases, so this data has 6000 cases.

I keep (x0, y0, z0) to (x7, y7, z7) features per cases, and drop (x8, y8, z8), (x9, y9, z9), (x10, y10, z10). I eliminate them because they don't have any numbers in class 3, all of them are missing value.

For continuous features kept in RDS: compute and display their mean and standard deviation within each class

each features' mean for class2:

52.18828	89.07367	-25.5622	54.27612	93.20038	-21.3277	52.32106	95.85241
-16.4069	50.28302	97.28839	-15.2335	47.21729	102.1962	-10.2448	48.64125
102.126	-11.4306	46.40612	102.1915	-11.4314	48.09056	100.0245	-14.1801

each features' standard deviation for class2:

35.60378	44.68207	40.1539	32.28403	45.83463	41.76554	31.74243	44.72356
41.30227	32.15475	45.43506	40.95418	32.55068	43.47915	39.11485	32.98501
44.54888	39.4683	32.90911	42.99767	39.31332	34.40838	44.6469	41.04382

each features' mean for class3:

59.85398	87.58127	-24.0744	64.52046	84.98173	-24.017	62.8638	77.1262
-29.0766	59.34622	72.88789	-32.027	56.70791	67.83633	-34.5118	53.13304

66.39769	-34.7836	49.43289	64.20236	-35.9666	17.69113	77.86752	-20.4511

each features' standard deviation for class3:

34.348	36.75472	25.36716	31.61421	40.03512	27.96902	35.79858	40.76816
31.33143	37.13003	41.62995	32.92938	36.94817	41.7094	33.22743	37.85801
41.79603	32.90039	38.74618	40.65076	31.9727	34.91182	32.10452	27.50974

each features' mean for class5:

41.80774	96.90318	-33.2649	27.26092	106.9172	-26.8532	22.33297	107.1237
-23.2646	22.23996	106.3067	-20.9532	21.72038	106.2558	-21.0432	22.33347
103.0138	-19.7606	23.16732	102.4325	-21.5186	27.38183	99.39649	-21.6192

each features' standard deviation for class5:

38.45318	41.02747	19.94227	38.1736	31.87687	21.01921	37.28126	29.87796
20.98247	37.1407	28.48001	20.92449	36.15452	28.81004	21.03925	38.09436
28.61621	21.11169	39.65073	29.07812	21.14757	42.14559	30.56257	21.37374

Step3: Center and Rescale the whole RDS so that each feature will then have global mean = 0 and global stand. dev. =1

Split each class into a training set and a test set, using the proportions 80% and 20%

the new sizes of the classes within TRAIN and within TEST

test_c2	400 obs. of 26 variables
test_c3	400 obs. of 26 variables
test_c5	400 obs. of 26 variables
train_c2	1600 obs. of 26 variables
train_c3	1600 obs. of 26 variables
train_c5	1600 obs. of 26 variables

the sizes of TRAIN and TEST

TRAIN	4800 obs. of 26 variables
TEST	1200 obs. of 26 variables

Because the performance is too perfect, I add some new cases into the test set.

The new test for each class and the whole test set:

test_c2	562 obs. of 26 variables
test_c3	569 obs. of 26 variables
test_c5	553 obs. of 26 variables
TEST	1684 obs. of 26 variables

Question 2: SVM classification by radial kernel

Step1: optimize the parameters "cost" and "gamma"

select CL2 and CL3 for the classification CL2 vs CL3

Select a list of 4 values for the "cost " parameter and a list of 4 values for the parameter "gamma": cost=c(0.1 ,1 ,10 ,100),gamma=c(0.1,1,10,100))

When the cost=10, gamma=0.1 we can get the best performance: error=0.0146875

```
Parameter tuning of ‘svm’:
- sampling method: 10-fold cross validation
- best parameters:
 cost gamma
   10   0.1
- best performance: 0.0146875
- Detailed performance results:
    cost gamma     error  dispersion
1    0.1   0.1 0.0506250 0.009637528
2    1.0   0.1 0.0162500 0.007336174
3   10.0   0.1 0.0146875 0.006917482
4  100.0   0.1 0.0146875 0.006917482
5    0.1   1.0 0.5312500 0.016204530
6    1.0   1.0 0.2971875 0.040355678
7   10.0   1.0 0.2762500 0.035909560
8  100.0   1.0 0.2762500 0.035909560
9    0.1  10.0 0.5312500 0.016204530
10   1.0  10.0 0.4337500 0.038578248
11  10.0  10.0 0.4325000 0.038617605
12 100.0  10.0 0.4325000 0.038617605
13   0.1 100.0 0.5312500 0.016204530
14   1.0 100.0 0.4909375 0.041756414
15  10.0 100.0 0.4896875 0.041196567
16 100.0 100.0 0.4896875 0.041196567
```

Step 2: Re-evaluation of tuning :

Pick another two classes CL2 vs CL5, with cost=c(0.1 ,1 ,10 ,100),gamma=c(0.1,1,10,100))

When the cost=1, gamma=0.1 we can get the best performance: error=0.0025

```
Parameter tuning of ‘svm’:
```

```
- sampling method: 10-fold cross validation
- best parameters:
 cost gamma
    1   0.1
- best performance: 0.0025
- Detailed performance results:
    cost gamma     error  dispersion
1    0.1   0.1 0.0221875 0.009931405
2    1.0   0.1 0.0025000 0.001976424
3   10.0   0.1 0.0025000 0.001976424
4  100.0   0.1 0.0025000 0.001976424
5    0.1   1.0 0.5312500 0.016204530
6    1.0   1.0 0.3975000 0.079927810
7   10.0   1.0 0.3721875 0.077804202
8  100.0   1.0 0.3721875 0.077804202
9    0.1  10.0 0.5312500 0.016204530
10   1.0  10.0 0.5225000 0.017292922
11  10.0  10.0 0.5225000 0.017292922
12 100.0  10.0 0.5225000 0.017292922
13   0.1 100.0 0.5312500 0.016204530
14   1.0 100.0 0.5240625 0.016864065
15  10.0 100.0 0.5240625 0.016864065
16 100.0 100.0 0.5240625 0.016864065
```

Fix the gamma=0.1, cost= 10

Question 3 : for the largest 3 classes CL1 CL2 CL3 , compute 3 SVMs

Use the best parameters previously identified to train 3 svms :

SVM1 to classify CL2 vs (not CL2)

```
Call:
svm(formula = TRAIN1$y ~ ., data = TRAIN1[3:26], kernel = "radial", ga
mma = 0.1, cost = 10, scale = FALSE)
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  10
Number of Support Vectors:  1703
 ( 828 875 )
```

```
Number of Classes:  2
Levels:
 -2 2
```

the percentages of support vectors for SVM1

1703/4800=35.48%

Confusion matrices for the training set:

```
     predict
real   -2    2
  -2 3200    0
  2     0 1600
```

Matrix in frequency of correct predictions within each class on training set:

	Real class: -2	Real class: 2
Predict class: -2	100%	0%
Predict class: 2	0%	100%

The correct prediction of PredTrain: 100%

Confusion matrices for the test set:

```
    predict
real   -2    2
  -2 1079   43
  2    10  552
```

Matrix in frequency of correct predictions within each class on test set:

	Real class: -2	Real class: 2
Predict class: -2	96.17%	1.78%
Predict class: 2	3.83%	98.22%

The correct prediction of PredTest is 96.85%

The errors of estimation on PredTEST:

$\sqrt{96.85\%(1-96.85\%)/1684} = 0.004$

95% confidence interval:

$96.85\% \pm 1.96 \times 0.004 = (96.07\%, 97.63\%)$

The errors of estimation on class(-2):

$\sqrt{96.17\%(1-96.17\%)/1122} = 0.006$

95% confidence interval:

$96.17\% \pm 1.96 \times 0.006 = (94.99\%, 97.35\%)$

The errors of estimation on class(2):

$\sqrt{98.22\%(1-98.22\%)/562} = 0.006$

95% confidence interval:

$98.22\% \pm 1.96 \times 0.006 =$ (97.04%,99.4%)

Interpretation:

The performance of SVM1 function with ‘kernel = radial’ and gamma=0.1, cost= 10 is good: the TRAIN set has 100% correct prediction and the 95% confidence interval for TEST set is (96.07%,97.63%).

Within each class, for both TRAIN and TEST set, the 95% confidence interval are on the right side of 95%.

So, the svm1 gives a very good performance for both TRAIN and TEST to determine if the posture that the users perform is 2=Stop (hand flat) or not.

SVM2 to classify CL3 vs (not CL3)

```
Call:
svm(formula = TRAIN2$y ~ ., data = TRAIN2[3:26], kernel = "radial", gamma = 0.1, cost = 10, scale = FALSE)
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  10
Number of Support Vectors:  1502
 ( 653 849 )
Number of Classes:  2
Levels:
 -3 3
```

the percentages of support vectors for SVM2

1502/4800=31.29%

Confusion matrices for the training set:

```
    predict
real   -3    3
  -3 3200    0
  3     0 1600
```

Matrix in frequency of correct predictions within each class on training set:

	Real class: -3	Real class: 3
Predict class: -3	100%	0%
Predict class: 3	0%	100%

The correct prediction of PredTrain: 100%

Confusion matrices for the test set:

```
   predict
real   -3    3
  -3 1109    6
  3    14  555
```

Matrix in frequency of correct predictions within each class on test set:

	Real class: -3	Real class: 3
Predict class: -3	99.46%	2.46%
Predict class: 3	0.54%	97.54%

The correct prediction of PredTest is 98.81%

The errors of estimation on PredTEST:

$\sqrt{98.81\%(1-98.81\%)/1684}$ = 0.003

95% confidence interval:

$98.81\% \pm 1.96 \times 0.003$ = (98.22%,99.4%)

The errors of estimation on class(-3):

$\sqrt{99.46\%(1-99.46\%)/1115}$ = 0.002

95% confidence interval:

$99.46\% \pm 1.96 \times 0.002$= (99.07%,99.85%)

The errors of estimation on class(3):

$\sqrt{97.54\%(1-97.54\%)/569}$ = 0.006

95% confidence interval:

$97.54\% \pm 1.96 \times 0.006$= (96.36%,98.72%)

Interpretation:

The performance of SVM2 function with ‘kernel = radial’ and gamma=0.1, cost= 10 is good: the TRAIN set has 100% correct prediction and the 95% confidence interval for TEST set is (98.22%,99.4%).

Within each class, for both TRAIN and TEST set, the 95% confidence interval are on the right side of 98%.

So, the svm2 gives a very good performance for both TRAIN and TEST to determine if the posture that the users perform is 3=Point1(point with pointer finger) or not.

SVM3 to classify CL5 vs (not CL5)

```
Call:
svm(formula = TRAIN3$y ~ ., data = TRAIN3[3:26], kernel = "radial", ga
mma = 0.1, cost = 10, scale = FALSE)
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  10
Number of Support Vectors:  1149
 ( 460 689 )
Number of Classes:  2
Levels:
 -5 5
```

the percentages of support vectors for SVM3

1583/4800=32.98%

Confusion matrices for the training set:

```
    predict
real   -5    5
  -5 3200    0
  5     0 1600
```

Matrix in frequency of correct predictions within each class on training set:

	Real class: -5	Real class: 5
Predict class: -5	100%	0%
Predict class: 5	0%	100%

The correct prediction of PredTrain: 100%

Confusion matrices for the test set:

```
    predict
real   -5    5
  -5 1128    3
  5    26  527
```

Matrix in frequency of correct predictions within each class on test set:

	Real class: -5	Real class: 5
Predict class: -5	99.73%	4.7%
Predict class: 5	0.27%	95.3%

The correct prediction of PredTest is 98.28%

The errors of estimation on PredTEST:

$\sqrt{98.28\%(1-98.28\%)/1684}$ = 0.003

95% confidence interval:

$98.28\% \pm 1.96 \times 0.003$ = (97.69%,98.87%)

The errors of estimation on class(-5):

$\sqrt{99.73\%(1-99.73\%)/1131}$ = 0.002

95% confidence interval:

$99.73\% \pm 1.96 \times 0.002$= (99.34%,100%)

The errors of estimation on class(5):

$\sqrt{95.3\%(1-95.3\%)/553}$ = 0.009

95% confidence interval:

$95.3\% \pm 1.96 \times 0.009$= (93.54%,97.06%)

Interpretation:

The performance of SVM3 function with 'kernel = radial' and gamma=0.1, cost= 10 is good: the TRAIN set has 100% correct prediction and the 95% confidence interval for TEST set is (97.69%,98.87%).

Within each class, for both TRAIN and TEST set, the 95% confidence interval are on the right side of 97%.

So, the svm3 gives a very good performance for both TRAIN and TEST to d etermine if the posture that the users perform is 5=Grab (fingers curled as if t o grab) or not.

Question 4 : for the largest 3 classes CL1 CL2 CL3 , combine the three SVMs to classify all cases

combined classification on all x in CL1 CL2 CL3 which belong to TRAIN

```
> tail(svm_train_pred)
     train_pred1 train_pred2 train_pred3 SVM1_rel2 SVM1_rel3 SVM1_rel5 SVM2_rel2 SVM2_rel3 SVM2_rel5 SVM3_rel2 SVM3_rel3 SVM3_rel5 score2 score3 score5 sum pred.CL reliability true.CL
5995          -2          -3           5         0       0.5       0.5       0.5         0       0.5         0         0         1    0.5    0.5      2   3     CL5   0.6666667       5
5996          -2          -3           5         0       0.5       0.5       0.5         0       0.5         0         0         1    0.5    0.5      2   3     CL5   0.6666667       5
5997          -2          -3           5         0       0.5       0.5       0.5         0       0.5         0         0         1    0.5    0.5      2   3     CL5   0.6666667       5
5998          -2          -3           5         0       0.5       0.5       0.5         0       0.5         0         0         1    0.5    0.5      2   3     CL5   0.6666667       5
5999          -2          -3           5         0       0.5       0.5       0.5         0       0.5         0         0         1    0.5    0.5      2   3     CL5   0.6666667       5
6000          -2          -3           5         0       0.5       0.5       0.5         0       0.5         0         0         1    0.5    0.5      2   3     CL5   0.6666667       5
```

Confusion matrices for the training set:

```
       real
predict    2    3    5
    CL2 1600    0    0
    CL3    0 1600    0
    CL5    0    0 1600
```

Matrix in frequency of correct predictions within each class on training set:

	Real class: 2	Real class: 3	Real class: 5
Predict class: 2	100%	0%	0%

Predict class: 3	0%	100%	0%
Predict class: 5	0%	0%	100%

The correct prediction of PredTrain: 100%

combined classification on all x in CL1 CL2 CL3 which belong to TEST

```
> tail(SVM_test_pred)
     test_pred1 test_pred2 test_pred3 SVM1_rel2 SVM1_rel3 SVM1_rel5 SVM2_rel2 SVM2_rel3 SVM2_rel5 SVM3_rel2 SVM3_rel3 SVM3_rel5 score2 score3 score5    sum pred.CL reliability true.CL
5949         -2         -3         -5         0    0.4911    0.4911    0.4877         0    0.4877    0.4765    0.4765    0.0000 0.9642 0.9676 0.9788 2.9106     CL5   0.3362881       5
5951         -2         -3          5         0    0.4911    0.4911    0.4877         0    0.4877    0.0000    0.0000    0.9973 0.4877 0.4911 1.9761 2.9549     CL5   0.6687536       5
5958         -2         -3          5         0    0.4911    0.4911    0.4877         0    0.4877    0.0000    0.0000    0.9973 0.4877 0.4911 1.9761 2.9549     CL5   0.6687536       5
5973         -2         -3          5         0    0.4911    0.4911    0.4877         0    0.4877    0.0000    0.0000    0.9973 0.4877 0.4911 1.9761 2.9549     CL5   0.6687536       5
5986         -2         -3          5         0    0.4911    0.4911    0.4877         0    0.4877    0.0000    0.0000    0.9973 0.4877 0.4911 1.9761 2.9549     CL5   0.6687536       5
5987         -2         -3          5         0    0.4911    0.4911    0.4877         0    0.4877    0.0000    0.0000    0.9973 0.4877 0.4911 1.9761 2.9549     CL5   0.6687536       5
```

Confusion matrices for the test set:

```
        real
predict   2   3   5
    CL2 495   5  17
    CL3  35 522   2
    CL5  32  42 534
```

Matrix in frequency of correct predictions within each class on test set:

	Real class: 2	Real class: 3	Real class: 5
Predict class: 2	88.08%	0.88%	3.07%
Predict class: 3	6.23%	91.74%	0.36%
Predict class: 5	5.69%	7.38%	96.57%

The correct prediction of PredTest is 92.1%

The errors of estimation on PredTEST:

$\sqrt{92.1\%(1-92.1\%)/1684} = 0.007$

95% confidence interval:

$92.1\% \pm 1.96 \times 0.007 =$ (90.73%,93.47%)

The errors of estimation on class(2):

$\sqrt{88.08\%(1-88.08\%)/562} = 0.014$

95% confidence interval:

$88.08\% \pm 1.96 \times 0.014 =$ (85.34%,90.82%)

The errors of estimation on class(3):

$\sqrt{91.74\%(1-91.74\%)/569} = 0.012$

95% confidence interval:

$91.74\% \pm 1.96 \times 0.012 =$ (89.39%,94.09%)

The errors of estimation on class(5):

$\sqrt{96.57\%(1-96.57\%)/553} = 0.008$

95% confidence interval:

$96.57\% \pm 1.96 \times 0.008 =$ (95%,98.14%)

Interpretation:

The performance of combined classification on all x in CL1 CL2 CL3 is good: the TRAIN set has 100% correct prediction and the 95% confidence interval for TEST set is (90.73%,93.47%).

Within each class, for both TRAIN and TEST set, the 95% confidence interval are on the right side of 90%.

So, the 3x3 matrices gives a very good performance for both TRAIN and TEST to determine what the posture that the users perform: 2=Stop (hand flat), 3= Point1(point with pointer finger), or 5=Grab (fingers curled as if to grab).

Question 5

select CL2 and CL3 for the classification CL2 vs CL3

Select a list of 5 values for the "cost " parameter cost=c(0.1 ,1 ,10 ,100 ,1000)

When the cost=100, we can get the best performance: error=0.0290625

```
Parameter tuning of 'svm':
- sampling method: 10-fold cross validation
- best parameters:
 cost
  100
- best performance: 0.0290625
- Detailed performance results:
   cost     error  dispersion
1 1e-01 0.0850000 0.015080801
2 1e+00 0.0390625 0.007254369
3 1e+01 0.0293750 0.008095566
4 1e+02 0.0290625 0.009205646
5 1e+03 0.0290625 0.009205646
```

Pick another two classes CL2 vs CL5, with cost=c(0.1 ,1 ,10 ,100 ,1000)

When the cost=10, we can get the best performance: error=0.0246875

```
Parameter tuning of 'svm':
- sampling method: 10-fold cross validation
- best parameters:
 cost
   10
- best performance: 0.0246875
- Detailed performance results:
   cost     error dispersion
```

```
1 1e-01 0.1253125 0.02116661
2 1e+00 0.0403125 0.01262524
3 1e+01 0.0246875 0.01066882
4 1e+02 0.0262500 0.00861503
5 1e+03 0.0262500 0.00861503
```

Fix cost = 10

Use the best parameters previously identified to train 3 svms :

SVM1 to classify CL2 vs (not CL2)

```
Call:
svm(formula = TRAIN1$y ~ ., data = TRAIN1[3:26], kernel = "polynomial
", coef0 = 1, degree = 2, cost = 10, scale = FALSE)
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  polynomial
       cost:  10
     degree:  2
     coef.0:  1
Number of Support Vectors:  268
 ( 113 155 )
Number of Classes:  2
Levels:
 -2 2
```

the percentages of support vectors for SVM1

268/4800=5.58%

Confusion matrices for the training set:

```
    predict
real   -2    2
  -2 3199    1
  2     2 1598
```

Matrix in frequency of correct predictions within each class on training set:

	Real class: -2	Real class: 2
Predict class: -2	99.97%	0.12%
Predict class: 2	0.03%	99.88%

The correct prediction of PredTrain: 99.94%

The errors of estimation on PredTRAIN:

$\sqrt{99.94\%(1-99.94\%)/4800} = 3.53 \times 10^{-4}$

95% confidence interval:

$99.94\% \pm 1.96 \times 3.53 \times 10^{-4} =$ (99.87%,100%)

The errors of estimation on class(-2):

$\sqrt{99.97\%(1-99.97\%)/3200} = 3.06 \times 10^{-4}$

95% confidence interval:

$99.97\% \pm 1.96 \times 3.06 \times 10^{-4}$ = (99.91%,100%)

The errors of estimation on class(2):

$\sqrt{99.88\%(1-99.88\%)/1600} = 8.66 \times 10^{-4}$

95% confidence interval:

$99.88\% \pm 1.96 \times 8.66 \times 10^{-4}$ = (99.71%,100%)

Confusion matrices for the test set:

```
    predict
real   -2    2
  -2 1065   57
  2    51  511
```

Matrix in frequency of correct predictions within each class on test set:

	Real class: -2	Real class: 2
Predict class: -2	94.92%	9.07%
Predict class: 2	5.08%	90.93%

The correct prediction of PredTest is 93.59%

The errors of estimation on PredTEST:

$\sqrt{93.59\%(1-93.59\%)/1684} = 5.97 \times 10^{-3}$

95% confidence interval:

$93.59\% \pm 1.96 \times 5.97 \times 10^{-3}$ = (92.42%,94.76%)

The errors of estimation on class(-2):

$\sqrt{94.92\%(1-94.92\%)/1122} = 6.56 \times 10^{-3}$

95% confidence interval:

$94.92\% \pm 1.96 \times 6.56 \times 10^{-3} =$ (93.63%,96.21%)

The errors of estimation on class(2):

$\sqrt{90.93\%(1-90.93\%)/562} = 0.012$

95% confidence interval:

$90.93\% \pm 1.96 \times 0.012 =$ (88.58%,93.28%)

Interpretation:

The performance of SVM1 function with 'kernel = polynomial' and cost= 10 is good: 95% confidence interval for TRAIN is (99.87%,100%) and 95% confidence interval for TEST is (92.42%,94.76%). Within each class, for both TRAIN and TEST set, the 95% confidence interval are on the right side of 88%.

So, the svm1 gives a very good performance for both TRAIN and TEST to determine if the posture that the users perform is 2=Stop (hand flat) or not.

SVM2 to classify CL3 vs (not CL3)

```
Call:
svm(formula = TRAIN2$y ~ ., data = TRAIN2[3:26], kernel = "polynomial
", coef0 = 1, degree = 2, cost = 10, scale = FALSE)
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  polynomial
       cost:  10
     degree:  2
     coef.0:  1
Number of Support Vectors:  285
 ( 127 158 )
Number of Classes:  2
Levels:
 -3 3
```

the percentages of support vectors for SVM2

285/4800=5.94%

Confusion matrices for the training set:

```
   predict
real   -3    3
  -3 3199    1
  3     4 1596
```

Matrix in frequency of correct predictions within each class on training set:

	Real class: -3	Real class: 3
Predict class: -3	99.97%	0.25%
Predict class: 3	0.03%	99.75%

The correct prediction of PredTrain: 99.9%

The errors of estimation on PredTRAIN:

$\sqrt{99.9\%(1-99.9\%)/4800} = 4.56 \times 10^{-4}$

95% confidence interval:

$99.9\% \pm 1.96 \times 4.56 \times 10^{-4}$ = (99.81%,99.99%)

The errors of estimation on class(-3):

$\sqrt{99.97\%(1-99.97\%)/3200}$ = 3.06×10^{-4}

95% confidence interval:

$99.97\% \pm 1.96 \times 3.06 \times 10^{-4}$ = (99.91%,100%)

The errors of estimation on class(3):

$\sqrt{99.75\%(1-99.75\%)/1600}$ = 1.25×10^{-3}

95% confidence interval:

$99.75\% \pm 1.96 \times 1.25 \times 10^{-3}$ = (99.51%,99.99%)

Confusion matrices for the test set:

```
      predict
real   -3    3
  -3 1110    5
  3     8  561
```

Matrix in frequency of correct predictions within each class on test set:

	Real class: -3	Real class: 3
Predict class: -3	99.55%	1.41%
Predict class: 3	0.45%	98.59%

The correct prediction of PredTest is 99.23%

The errors of estimation on PredTEST:

$\sqrt{99.23\%(1-99.23\%)/1684}$ = 2.13×10^{-3}

95% confidence interval:

$99.23\% \pm 1.96 \times 2.13 \times 10^{-3}$ = (98.81%,99.65%)

The errors of estimation on class(-3):

$\sqrt{99.55\%(1-99.55\%)/1115}$ = 0.002

95% confidence interval:

$99.55\% \pm 1.96 \times 0.002$= (99.16%,99.94%)

The errors of estimation on class(3):

$\sqrt{98.59\%(1-98.59\%)/569}$ = 4.94×10^{-3}

95% confidence interval:

$98.59\% \pm 1.96 \times 4.94 \times 10^{-3}$ = (97.62%,99.56%)

Interpretation:

The performance of SVM2 function with ‘kernel = polynomial’ and cost= 10 is good: 95% confidence interval for TRAIN is (99.81%,99.99%) and the 95% confidence interval for TEST set is (98.81%,99.65%).

Within each class, for both TRAIN and TEST set, the 95% confidence interval are on the right side of 97%.
So, the svm2 gives a very good performance for both TRAIN and TEST to determine if the posture that the users perform is 3=Point1(point with pointer finger) or not.

SVM3 to classify CL5 vs (not CL5)

```
Call:
svm(formula = TRAIN3$y ~ ., data = TRAIN3[3:26], kernel = "polynomial
", coef0 = 1, degree = 2, cost = 10, scale = FALSE)
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  polynomial
       cost:  10
     degree:  2
     coef.0:  1
Number of Support Vectors:  209
 ( 104 105 )
Number of Classes:  2
Levels:
 -5 5
```

the percentages of support vectors for SVM3
209/4800=4.35%

Confusion matrices for the training set:

```
    predict
real   -5    5
  -5 3200    0
  5     0 1600
```

Matrix in frequency of correct predictions within each class on training set:

	Real class: -5	Real class: 5
Predict class: -5	100%	0%
Predict class: 5	0%	100%

The correct prediction of PredTrain: 100%

Confusion matrices for the test set:

```
    predict
real   -5    5
  -5 1128    3
  5     2  551
```

Matrix in frequency of correct predictions within each class on test set:

	Real class: -5	Real class: 5
Predict class: -5	99.73%	0.36%
Predict class: 5	0.27%	99.64%

The correct prediction of PredTest is 99.7%

The errors of estimation on PredTEST:

$\sqrt{99.7\%(1-99.7\%)/1684} = 1.33\times 10^{-3}$

95% confidence interval:

$99.7\% \pm 1.96 \times 1.33 \times 10^{-3} = (99.44\%, 99.96\%)$

The errors of estimation on class(-5):

$\sqrt{99.73\%(1-99.73\%)/1131} = 1.54\times 10^{-3}$

95% confidence interval:

$99.73\% \pm 1.96 \times 1.54\times 10^{-3} = (99.43\%, 100\%)$

The errors of estimation on class(5):

$\sqrt{99.64\%(1-99.64\%)/553} = 2.55\times 10^{-3}$

95% confidence interval:

$99.64\% \pm 1.96 \times 2.55 \times 10^{-3} = (99.14\%, 100\%)$

Interpretation:

The performance of SVM3 function with 'kernel = polynomial' and cost= 10 is good: the TRAIN set has 100% correct prediction and the 95% confidence interval for TEST set is (99.44%,99.96%).

Within each class, for both TRAIN and TEST set, the 95% confidence interval are on the right side of 99%.

So, the svm3 gives a very good performance for both TRAIN and TEST to d etermine if the posture that the users perform is 5=Grab (fingers curled as if t o grab) or not.

combined classification on all x in CL1 CL2 CL3 which belong to TRAIN

```
> tail(SVM_train_pred.1)
     train_pred1.1 train_pred2.1 train_pred3.1 SVM1_rel2 SVM1_rel3 SVM1_rel5 SVM2_rel2 SVM2_rel3 SVM2_rel5 SVM3_rel2 SVM3_rel3 SVM3_rel5  score2 score3  score5    sum pred.CL reliability true.CL
5995            -2            -3             5         0    0.4994    0.4994   0.49875         0   0.49875         0         0         1 0.49875 0.4994 1.99815 2.9963     CL5   0.6668725       5
5996            -2            -3             5         0    0.4994    0.4994   0.49875         0   0.49875         0         0         1 0.49875 0.4994 1.99815 2.9963     CL5   0.6668725       5
5997            -2            -3             5         0    0.4994    0.4994   0.49875         0   0.49875         0         0         1 0.49875 0.4994 1.99815 2.9963     CL5   0.6668725       5
5998            -2            -3             5         0    0.4994    0.4994   0.49875         0   0.49875         0         0         1 0.49875 0.4994 1.99815 2.9963     CL5   0.6668725       5
5999            -2            -3             5         0    0.4994    0.4994   0.49875         0   0.49875         0         0         1 0.49875 0.4994 1.99815 2.9963     CL5   0.6668725       5
6000            -2            -3             5         0    0.4994    0.4994   0.49875         0   0.49875         0         0         1 0.49875 0.4994 1.99815 2.9963     CL5   0.6668725       5
```

Confusion matrices for the training set:

```
        real
predict    2    3    5
    CL2 1598    0    0
    CL3    2 1599    0
    CL5    0    0 1600
```

Matrix in frequency of correct predictions within each class on training set:

	Real class: 2	Real class: 3	Real class: 5
Predict class: 2	99.88%	0%	0%
Predict class: 3	0.12%	100%	0%
Predict class: 5	0%	0%	100%

The correct prediction of PredTrain: 99.94%

The errors of estimation on PredTrain:

$\sqrt{99.94\%(1-99.94\%)/4800} = 3.53\times 10^{-4}$

95% confidence interval:

$99.94\% \pm 1.96 \times 3.53 \times 10^{-4} =$ (99.87%,100%)

The errors of estimation on class(2):

$\sqrt{99.88\%(1-99.88\%)/1600} = 8.66\times 10^{-4}$

95% confidence interval:

$99.88\% \pm 1.96 \times 8.66 \times 10^{-4} =$ (99.71%,100%)

combined classification on all x in CL1 CL2 CL3 which belong to TEST

```
> tail(SVM_test_pred.1)
     test_pred1.1 test_pred2.1 test_pred3.1 SVM1_rel2 SVM1_rel3 SVM1_rel5 SVM2_rel2 SVM2_rel3 SVM2_rel5 SVM3_rel2 SVM3_rel3 SVM3_rel5  score2  score3 score5    sum pred.CL reliability true.CL
5949           -2           -3            5         0   0.45465   0.45465   0.49295         0   0.49295         0         0    0.9973 0.49295 0.45465 1.9449 2.8925     CL5   0.6723941       5
5951           -2           -3            5         0   0.45465   0.45465   0.49295         0   0.49295         0         0    0.9973 0.49295 0.45465 1.9449 2.8925     CL5   0.6723941       5
5958           -2           -3            5         0   0.45465   0.45465   0.49295         0   0.49295         0         0    0.9973 0.49295 0.45465 1.9449 2.8925     CL5   0.6723941       5
5973           -2           -3            5         0   0.45465   0.45465   0.49295         0   0.49295         0         0    0.9973 0.49295 0.45465 1.9449 2.8925     CL5   0.6723941       5
5986           -2           -3            5         0   0.45465   0.45465   0.49295         0   0.49295         0         0    0.9973 0.49295 0.45465 1.9449 2.8925     CL5   0.6723941       5
5987           -2           -3            5         0   0.45465   0.45465   0.49295         0   0.49295         0         0    0.9973 0.49295 0.45465 1.9449 2.8925     CL5   0.6723941       5
```

Confusion matrices for the test set:

```
        real
predict   2   3   5
    CL2 497  41  28
    CL3  35 527   2
    CL5  30   1 523
```

Matrix in frequency of correct predictions within each class on test set:

	Real class: 2	Real class: 3	Real class: 5
Predict class: 2	88.43%	7.21%	5.06%
Predict class: 3	6.23%	92.62%	0.36%
Predict class: 5	5.34%	0.17%	94.58%

The correct prediction of PredTest is 91.86%

The errors of estimation on PredTEST:

$\sqrt{91.86\%(1-91.86\%)/1684} = 0.007$

95% confidence interval:

$91.86\% \pm 1.96 \times 0.007 =$ (90.49%,93.23%)

The errors of estimation on class(2):

$\sqrt{88.43\%(1-88.43\%)/562} = 0.013$

95% confidence interval:

$88.43\% \pm 1.96 \times 0.013 =$ (85.88%,90.98%)

The errors of estimation on class(3):

$\sqrt{92.62\%(1-92.62\%)/569} = 0.011$

95% confidence interval:

$92.62\% \pm 1.96 \times 0.011 =$ (90.46%,94.78%)

The errors of estimation on class(5):

$\sqrt{94.58\%(1-94.58\%)/553} = 0.01$

95% confidence interval:

$94.58\% \pm 1.96 \times 0.01 =$ (92.62%,96.54%)

Interpretation:

The performance of combined classification on all x in CL1 CL2 CL3 is good: the 95% confidence interval for TRAIN set is (99.87%,100%) and the 95% confidence interval for TEST set is (90.49%,93.23%).

Within each class, for both TRAIN and TEST set, the 95% confidence interval are on the right side of 85%.

So, the 3x3 matrices gives a very good performance for both TRAIN and TEST to determine what the posture that the users perform: 2=Stop (hand flat), 3= Point1(point with pointer finger), or 5=Grab (fingers curled as if to grab).

Both of the radial kernel and polynomial kernel gives a very good performance.

The performance of radial kernel is a little bit better than the polynomial kernel.

Code:

```
data=read.csv("C:/Users/yingy/Desktop/11 data mining/hw4/data.csv")
colnames(data)[1] <- "Class"
attach(data)

#for continuous features kept in RDS:compute and display their mean and standard
```

deviation within each class

```
c2=data[which(data$Class=="2"),]
c3=data[which(data$Class=="3"),]
c5=data[which(data$Class=="5"),]
#compute mean and sd in c2
for (i in 3:26){
   print(mean(c2[,i]))
}
for (i in 3:26){
   print(sd(c2[,i]))
}

#compute mean and sd in c3
for (i in 3:26){
   print(mean(c3[,i]))
}
for (i in 3:26){
   print(sd(c3[,i]))
}

#compute mean and sd in c5
for (i in 3:26){
   print(mean(c5[,i]))
}
for (i in 3:26){
   print(sd(c5[,i]))
}
#Step3 : Center and Rescale the whole RDS so that each feature will then have global
mean = 0 and global stand. dev. =1
library(scales)
cen_data <- scale(data[,3:26])
cen_data <- data.frame(cen_data)
new_data <- cbind(data[,1:2],cen_data)
new_data = data.frame(new_data)
c2=new_data[which(new_data$Class=="2"),]
c3=new_data[which(new_data$Class=="3"),]
c5=new_data[which(new_data$Class=="5"),]
```

```
##add more case in test
addtest=read.csv("C:/Users/yingy/Desktop/11 data mining/hw4/test.csv")
colnames(addtest)[1] <- "Class"
attach(addtest)
cen_addtest <- scale(addtest[,3:26])
cen_addtest <- data.frame(cen_addtest)
new_addtest <- cbind(addtest[,1:2],cen_addtest)
new_addtest = data.frame(new_addtest)
c2add=new_addtest[which(new_addtest$Class=="2"),]
c3add=new_addtest[which(new_addtest$Class=="3"),]
c5add=new_addtest[which(new_addtest$Class=="5"),]



#Split each class into a training set and a test set , using the proportions 80% and
20%
n <- 2000    # Number of observations
ntrain <- round(n*0.8)    # 80% for training set
set.seed(314)        # Set seed for reproducible results
tindex <- sample(n, ntrain)      # Create a random index
train_c2 <- c2[tindex,]      # Create c2 training set
test_c2 <- c2[-tindex,]      # Create c2 test set
test_c2 <- rbind(test_c2,c2add)
train_c3 <- c3[tindex,]      # Create c3 training set
test_c3 <- c3[-tindex,]      # Create c3 test set
test_c3 <- rbind(test_c3,c3add)
train_c5 <- c5[tindex,]      # Create c5 training set
test_c5 <- c5[-tindex,]      # Create c5 test set
test_c5 <- rbind(test_c5,c5add)

TRAIN <- rbind(train_c2,train_c3,train_c5)
TEST <- rbind(test_c2,test_c3,test_c5)

#Question 2: SVM classification by radial kernel
#Step1: optimize the parameters "cost" and "gamma"
x1=rbind(train_c2,train_c3)
```

```
data1=data.frame(x=x1[3:26],y=as.factor(x1$Class))

set.seed (1)
library(e1071)
tune.out1=tune(svm ,y~ .,data=data1,kernel ="radial",ranges
=list(cost=c(0.1 ,1 ,10 ,100),gamma=c(0.1,1,10,100) ))
summary (tune.out1)


#Step 2: Re-evaluation of tuning :
x2=rbind(train_c2,train_c5)
data2=data.frame(x=x2[3:26],y=as.factor(x2$Class))

set.seed (1)
tune.out2=tune(svm ,y~ .,data=data2,kernel ="radial",ranges
=list(cost=c(0.1 ,1 ,10 ,100),gamma=c(0.1,1,10,100) ))
summary (tune.out2)

#Question 3 : for the largest 3 classes CL1 CL2 CL3 , compute 3 SVMs
#SVM1 to classify CL2 vs (not CL2)
c35=rbind(train_c3,train_c5)
c35[,1] <- "-2"
TRAIN1=rbind(train_c2,c35)
library(e1071)
TRAIN1$y=as.factor(TRAIN1$Class)
svmfit1=svm(TRAIN1$y~ .,data=TRAIN1[3:26],kernel="radial",gamma
=0.1,cost=10,scale=FALSE)
summary(svmfit1)

c35_test=rbind(test_c3,test_c5)
c35_test[,1] <- "-2"
TEST1=rbind(test_c2,c35_test)
TEST1$y=as.factor(TEST1$Class)

#the percentages of correct predictions PredTrain and PredTest and the two
confusion matrices
train_pred1 = predict(svmfit1, TRAIN1[3:26])
test_pred1 = predict(svmfit1, TEST1[3:26])
```

```
#confusion matrices for TRAIN
TRAIN_confus.matrix1 = table(real=TRAIN1$y, predict=train_pred1)
TRAIN_confus.matrix1
#confusion matrices for TEST
TEST_confus.matrix1 = table(real=TEST1$y, predict=test_pred1)
TEST_confus.matrix1
#compute the errors of estimation on PredTRAIN, PredTEST, and on the terms of the
confusion matrices
sum(diag(TRAIN_confus.matrix1))/sum(TRAIN_confus.matrix1)
sum(diag(TEST_confus.matrix1))/sum(TEST_confus.matrix1)

#SVM2 to classify CL3 vs (not CL3)
c25=rbind(train_c2,train_c5)
c25[,1] <- "-3"
TRAIN2=rbind(train_c3,c25)
TRAIN2$y=as.factor(TRAIN2$Class)
svmfit2=svm(TRAIN2$y~ .,data=TRAIN2[3:26],kernel="radial",gamma
=0.1,cost=10,scale=FALSE)
summary(svmfit2)

c25_test=rbind(test_c2,test_c5)
c25_test[,1] <- "-3"
TEST2=rbind(test_c3,c25_test)
TEST2$y=as.factor(TEST2$Class)

#the percentages of correct predictions PredTrain and PredTest and the two
confusion matrices
train_pred2 = predict(svmfit2, TRAIN2[3:26])
test_pred2 = predict(svmfit2, TEST2[3:26])

#confusion matrices for TRAIN
TRAIN_confus.matrix2 = table(real=TRAIN2$y, predict=train_pred2)
TRAIN_confus.matrix2
#confusion matrices for TEST
TEST_confus.matrix2 = table(real=TEST2$y, predict=test_pred2)
TEST_confus.matrix2
```

```
#SVM3 to classify CL5 vs (not CL5)
c23=rbind(train_c2,train_c3)
c23[,1] <- "-5"
TRAIN3=rbind(train_c5,c23)
TRAIN3$y=as.factor(TRAIN3$Class)
svmfit3=svm(TRAIN3$y~ .,data=TRAIN3[3:26],kernel="radial",gamma
=0.1,cost=10,scale=FALSE)
summary(svmfit3)

c23_test=rbind(test_c2,test_c3)
c23_test[,1] <- "-5"
TEST3=rbind(test_c5,c23_test)
TEST3$y=as.factor(TEST3$Class)

#the percentages of correct predictions PredTrain and PredTest and the two
confusion matrices
train_pred3 = predict(svmfit3, TRAIN3[3:26])
test_pred3 = predict(svmfit3, TEST3[3:26])

#confusion matrices for TRAIN
TRAIN_confus.matrix3 = table(real=TRAIN3$y, predict=train_pred3)
TRAIN_confus.matrix3
#confusion matrices for TEST
TEST_confus.matrix3 = table(real=TEST3$y, predict=test_pred3)
TEST_confus.matrix3


#Q4:for the largest 3 classes CL1 CL2 CL3 , combine the three SVMs to classify all
cases
train_pred1= data.frame(train_pred1)
train_pred2= data.frame(train_pred2)
train_pred3= data.frame(train_pred3)
train_pred1=train_pred1[order(as.numeric(rownames(train_pred1))),,drop=FALSE]
train_pred2=train_pred2[order(as.numeric(rownames(train_pred2))),,drop=FALSE]
train_pred3=train_pred3[order(as.numeric(rownames(train_pred3))),,drop=FALSE]
SVM_train_pred <-    cbind(train_pred1,train_pred2,train_pred3)
#head(SVM_train_pred)
#tail(SVM_train_pred)
```

```
SVM_train_pred$SVM1_rel2 <- ifelse(SVM_train_pred$train_pred1 =='2', 1,0)
SVM_train_pred$SVM1_rel3 <- ifelse(SVM_train_pred$train_pred1 =='2',0, 1/2)
SVM_train_pred$SVM1_rel5 <- ifelse(SVM_train_pred$train_pred1 =='2',0, 1/2)

SVM_train_pred$SVM2_rel2 <- ifelse(SVM_train_pred$train_pred2 =='3', 0, 1/2)
SVM_train_pred$SVM2_rel3 <- ifelse(SVM_train_pred$train_pred2 =='3', 1, 0)
SVM_train_pred$SVM2_rel5 <- ifelse(SVM_train_pred$train_pred2 =='3', 0, 1/2)

SVM_train_pred$SVM3_rel2 <- ifelse(SVM_train_pred$train_pred3 =='5', 0, 1/2)
SVM_train_pred$SVM3_rel3 <- ifelse(SVM_train_pred$train_pred3 =='5', 0, 1/2)
SVM_train_pred$SVM3_rel5 <- ifelse(SVM_train_pred$train_pred3 =='5', 1, 0)


SVM_train_pred$score2 <-
SVM_train_pred$SVM1_rel2+SVM_train_pred$SVM2_rel2+SVM_train_pred$SVM3_r
el2
SVM_train_pred$score3 <-
SVM_train_pred$SVM1_rel3+SVM_train_pred$SVM2_rel3+SVM_train_pred$SVM3_r
el3
SVM_train_pred$score5 <-
SVM_train_pred$SVM1_rel5+SVM_train_pred$SVM2_rel5+SVM_train_pred$SVM3_r
el5

SVM_train_pred$sum <-
SVM_train_pred$score2+SVM_train_pred$score3+SVM_train_pred$score5

SVM_train_pred$pred.CL <- ifelse(SVM_train_pred$score2 > SVM_train_pred$score3
& SVM_train_pred$score2 > SVM_train_pred$score5,"CL2",
                                        ifelse(SVM_train_pred$score3 >
SVM_train_pred$score2 & SVM_train_pred$score3 > SVM_train_pred$score5,"CL3" ,
                                                 ifelse(SVM_train_pred$score5 >
SVM_train_pred$score2 & SVM_train_pred$score5 > SVM_train_pred$score3,"CL5",
NA)))

SVM_train_pred$reliability <- ifelse(SVM_train_pred$pred.CL == "CL2",
SVM_train_pred$score2/SVM_train_pred$sum,
                                             ifelse(SVM_train_pred$pred.CL ==
"CL3", SVM_train_pred$score3/SVM_train_pred$sum,
```

```
ifelse(SVM_train_pred$pred.CL == "CL5",
SVM_train_pred$score5/SVM_train_pred$sum,NA)))

TRAIN=TRAIN[order(as.numeric(rownames(TRAIN))),,drop=FALSE]
train_labels <- TRAIN[,1]
SVM_train_pred$true.CL <- train_labels

tail(SVM_train_pred)
table(predict=SVM_train_pred$pred.CL, real=SVM_train_pred$true.CL)



##test
test_pred1 <- data.frame(test_pred1)
test_pred2 <- data.frame(test_pred2)
test_pred3 <- data.frame(test_pred3)
test_pred1=test_pred1[order(as.numeric(rownames(test_pred1))),,drop=FALSE]
test_pred2=test_pred2[order(as.numeric(rownames(test_pred2))),,drop=FALSE]
test_pred3=test_pred3[order(as.numeric(rownames(test_pred3))),,drop=FALSE]

SVM_test_pred <-    cbind(test_pred1,test_pred2,test_pred3)

SVM_test_pred$SVM1_rel2 <- ifelse(SVM_test_pred$test_pred1 =='2', 1*0.9617,0)
SVM_test_pred$SVM1_rel3 <- ifelse(SVM_test_pred$test_pred1 =='2',0, 0.5*0.9822)
SVM_test_pred$SVM1_rel5 <- ifelse(SVM_test_pred$test_pred1 =='2',0, 0.5*0.9822)

SVM_test_pred$SVM2_rel2 <- ifelse(SVM_test_pred$test_pred2 =='3', 0,
0.5*0.9754)
SVM_test_pred$SVM2_rel3 <- ifelse(SVM_test_pred$test_pred2 =='3', 1*0.9946, 0)
SVM_test_pred$SVM2_rel5 <- ifelse(SVM_test_pred$test_pred2 =='3', 0,
0.5*0.9754)

SVM_test_pred$SVM3_rel2 <- ifelse(SVM_test_pred$test_pred3 =='5', 0, 0.5*0.953)
SVM_test_pred$SVM3_rel3 <- ifelse(SVM_test_pred$test_pred3 =='5', 0, 0.5*0.953)
SVM_test_pred$SVM3_rel5 <- ifelse(SVM_test_pred$test_pred3 =='5', 1*0.9973, 0)

SVM_test_pred$score2 <-
```

```
SVM_test_pred$SVM1_rel2+SVM_test_pred$SVM2_rel2+SVM_test_pred$SVM3_rel
2
SVM_test_pred$score3 <-
SVM_test_pred$SVM1_rel3+SVM_test_pred$SVM2_rel3+SVM_test_pred$SVM3_rel
3
SVM_test_pred$score5 <-
SVM_test_pred$SVM1_rel5+SVM_test_pred$SVM2_rel5+SVM_test_pred$SVM3_rel
5

SVM_test_pred$sum <-
SVM_test_pred$score2+SVM_test_pred$score3+SVM_test_pred$score5

SVM_test_pred$pred.CL <- ifelse(SVM_test_pred$score2 > SVM_test_pred$score3 &
SVM_test_pred$score2 > SVM_test_pred$score5,"CL2",
                                                    ifelse(SVM_test_pred$score3 >
SVM_test_pred$score2 & SVM_test_pred$score3 > SVM_test_pred$score5,"CL3" ,
                                                               ifelse(SVM_test_pred$score5 >
SVM_test_pred$score2 & SVM_test_pred$score5 > SVM_test_pred$score3,"CL5",
NA)))

SVM_test_pred$reliability <- ifelse(SVM_test_pred$pred.CL == "CL2",
SVM_test_pred$score2/SVM_test_pred$sum,
                                                         ifelse(SVM_test_pred$pred.CL == "CL3",
SVM_test_pred$score3/SVM_test_pred$sum,
                                                                    ifelse(SVM_test_pred$pred.CL
== "CL5", SVM_test_pred$score5/SVM_test_pred$sum,NA)))

TEST=TEST[order(as.numeric(rownames(TEST))),,drop=FALSE]
test_labels <- TEST[,1]
SVM_test_pred$true.CL <- test_labels

tail(SVM_test_pred)
table(predict=SVM_test_pred$pred.CL, real=SVM_test_pred$true.CL)

#Q5:using the polynomial kernel (K(x,y) = (1+<x,y>)2
#optimize the parameters "cost"
set.seed (1)
tune.out1.1=tune(svm ,y~ .,data=data1,kernel ="polynomial",ranges
```

```
=list(cost=c(0.1 ,1 ,10 ,100 ,1000)))
summary (tune.out1.1)


set.seed (1)
tune.out2.1=tune(svm ,y~ .,data=data2,kernel ="polynomial",ranges
=list(cost=c(0.1 ,1 ,10 ,100 ,1000)))
summary (tune.out2.1)

#SVM1 to classify CL2 vs (not CL2)
svmfit1.1=svm(TRAIN1$y~ .,data=TRAIN1[3:26],kernel="polynomial",coef0=1,degree
=2,cost=10,scale=FALSE)
summary(svmfit1.1)

#the percentages of correct predictions PredTrain and PredTest and the two
confusion matrices
train_pred1.1 = predict(svmfit1.1, TRAIN1[3:26])
test_pred1.1 = predict(svmfit1.1, TEST1[3:26])

#confusion matrices for TRAIN
TRAIN_confus.matrix1.1 = table(real=TRAIN1$y, predict=train_pred1.1)
TRAIN_confus.matrix1.1
#confusion matrices for TEST
TEST_confus.matrix1.1 = table(real=TEST1$y, predict=test_pred1.1)
TEST_confus.matrix1.1

#SVM2 to classify CL3 vs (not CL3)
svmfit2.1=svm(TRAIN2$y~ .,data=TRAIN2[3:26],kernel="polynomial",coef0=1,degree
=2,cost=10,scale=FALSE)
summary(svmfit2.1)

#the percentages of correct predictions PredTrain and PredTest and the two
confusion matrices
train_pred2.1 = predict(svmfit2.1, TRAIN2[3:26])
test_pred2.1 = predict(svmfit2.1, TEST2[3:26])

#confusion matrices for TRAIN
TRAIN_confus.matrix2.1 = table(real=TRAIN2$y, predict=train_pred2.1)
```

```
TRAIN_confus.matrix2.1
#confusion matrices for TEST
TEST_confus.matrix2.1 = table(real=TEST2$y, predict=test_pred2.1)
TEST_confus.matrix2.1

#SVM3 to classify CL5 vs (not CL5)
svmfit3.1=svm(TRAIN3$y~ .,data=TRAIN3[3:26],kernel="polynomial",coef0=1,degree
=2,cost=10,scale=FALSE)
summary(svmfit3.1)

#the percentages of correct predictions PredTrain and PredTest and the two
confusion matrices
train_pred3.1 = predict(svmfit3.1, TRAIN3[3:26])
test_pred3.1 = predict(svmfit3.1, TEST3[3:26])

#confusion matrices for TRAIN
TRAIN_confus.matrix3.1 = table(real=TRAIN3$y, predict=train_pred3.1)
TRAIN_confus.matrix3.1
#confusion matrices for TEST
TEST_confus.matrix3.1 = table(real=TEST3$y, predict=test_pred3.1)
TEST_confus.matrix3.1


#train
train_pred1.1= data.frame(train_pred1.1)
train_pred2.1= data.frame(train_pred2.1)
train_pred3.1= data.frame(train_pred3.1)
train_pred1.1=train_pred1.1[order(as.numeric(rownames(train_pred1.1))),,drop=FAL
SE]
train_pred2.1=train_pred2.1[order(as.numeric(rownames(train_pred2.1))),,drop=FAL
SE]
train_pred3.1=train_pred3.1[order(as.numeric(rownames(train_pred3.1))),,drop=FAL
SE]
SVM_train_pred.1 <-    cbind(train_pred1.1,train_pred2.1,train_pred3.1)
#head(SVM_train_pred.1)
#tail(SVM_train_pred.1)
SVM_train_pred.1$SVM1_rel2 <- ifelse(SVM_train_pred.1$train_pred1.1 =='2',
1*0.9997,0)
```

```
SVM_train_pred.1$SVM1_rel3 <- ifelse(SVM_train_pred.1$train_pred1.1 =='2',0,
0.5*0.9988)
SVM_train_pred.1$SVM1_rel5 <- ifelse(SVM_train_pred.1$train_pred1.1 =='2',0,
0.5*0.9988)

SVM_train_pred.1$SVM2_rel2 <- ifelse(SVM_train_pred.1$train_pred2.1 =='3', 0,
0.5*0.9975)
SVM_train_pred.1$SVM2_rel3 <- ifelse(SVM_train_pred.1$train_pred2.1 =='3',
1*0.9997, 0)
SVM_train_pred.1$SVM2_rel5 <- ifelse(SVM_train_pred.1$train_pred2.1 =='3', 0,
0.5*0.9975)

SVM_train_pred.1$SVM3_rel2 <- ifelse(SVM_train_pred.1$train_pred3.1 =='5', 0,
1/2)
SVM_train_pred.1$SVM3_rel3 <- ifelse(SVM_train_pred.1$train_pred3.1 =='5', 0,
1/2)
SVM_train_pred.1$SVM3_rel5 <- ifelse(SVM_train_pred.1$train_pred3.1 =='5', 1, 0)


SVM_train_pred.1$score2 <-
SVM_train_pred.1$SVM1_rel2+SVM_train_pred.1$SVM2_rel2+SVM_train_pred.1$S
VM3_rel2
SVM_train_pred.1$score3 <-
SVM_train_pred.1$SVM1_rel3+SVM_train_pred.1$SVM2_rel3+SVM_train_pred.1$S
VM3_rel3
SVM_train_pred.1$score5 <-
SVM_train_pred.1$SVM1_rel5+SVM_train_pred.1$SVM2_rel5+SVM_train_pred.1$S
VM3_rel5

SVM_train_pred.1$sum <-
SVM_train_pred.1$score2+SVM_train_pred.1$score3+SVM_train_pred.1$score5

SVM_train_pred.1$pred.CL <- ifelse(SVM_train_pred.1$score2 >
SVM_train_pred.1$score3 & SVM_train_pred.1$score2 >
SVM_train_pred.1$score5,"CL2",
                                   ifelse(SVM_train_pred.1$score3 >
SVM_train_pred.1$score2 & SVM_train_pred.1$score3 >
SVM_train_pred.1$score5,"CL3" ,
```

```
                                                      ifelse(SVM_train_pred.1$score5
> SVM_train_pred.1$score2 & SVM_train_pred.1$score5 >
SVM_train_pred.1$score3,"CL5", NA)))

SVM_train_pred.1$reliability <- ifelse(SVM_train_pred.1$pred.CL == "CL2",
SVM_train_pred.1$score2/SVM_train_pred.1$sum,
                                                  ifelse(SVM_train_pred.1$pred.CL ==
"CL3", SVM_train_pred.1$score3/SVM_train_pred.1$sum,

ifelse(SVM_train_pred.1$pred.CL == "CL5",
SVM_train_pred.1$score5/SVM_train_pred.1$sum,NA)))

SVM_train_pred.1$true.CL <- train_labels

tail(SVM_train_pred.1)
table(predict=SVM_train_pred.1$pred.CL, real=SVM_train_pred.1$true.CL)



#test
test_pred1.1= data.frame(test_pred1.1)
test_pred2.1= data.frame(test_pred2.1)
test_pred3.1= data.frame(test_pred3.1)
test_pred1.1=test_pred1.1[order(as.numeric(rownames(test_pred1.1))),,drop=FALSE
]
test_pred2.1=test_pred2.1[order(as.numeric(rownames(test_pred2.1))),,drop=FALSE
]
test_pred3.1=test_pred3.1[order(as.numeric(rownames(test_pred3.1))),,drop=FALSE
]
SVM_test_pred.1 <-    cbind(test_pred1.1,test_pred2.1,test_pred3.1)
#head(SVM_test_pred.1)
#tail(SVM_test_pred.1)
SVM_test_pred.1$SVM1_rel2 <- ifelse(SVM_test_pred.1$test_pred1.1 =='2',
1*0.9492,0)
SVM_test_pred.1$SVM1_rel3 <- ifelse(SVM_test_pred.1$test_pred1.1 =='2',0,
0.5*0.9093)
SVM_test_pred.1$SVM1_rel5 <- ifelse(SVM_test_pred.1$test_pred1.1 =='2',0,
0.5*0.9093)
```

```
SVM_test_pred.1$SVM2_rel2 <- ifelse(SVM_test_pred.1$test_pred2.1 =='3', 0,
0.5*0.9859)
SVM_test_pred.1$SVM2_rel3 <- ifelse(SVM_test_pred.1$test_pred2.1 =='3',
1*0.9955, 0)
SVM_test_pred.1$SVM2_rel5 <- ifelse(SVM_test_pred.1$test_pred2.1 =='3', 0,
0.5*0.9859)

SVM_test_pred.1$SVM3_rel2 <- ifelse(SVM_test_pred.1$test_pred3.1 =='5', 0,
0.5*0.9964)
SVM_test_pred.1$SVM3_rel3 <- ifelse(SVM_test_pred.1$test_pred3.1 =='5', 0,
0.5*0.9964)
SVM_test_pred.1$SVM3_rel5 <- ifelse(SVM_test_pred.1$test_pred3.1 =='5',
1*0.9973, 0)


SVM_test_pred.1$score2 <-
SVM_test_pred.1$SVM1_rel2+SVM_test_pred.1$SVM2_rel2+SVM_test_pred.1$SVM
3_rel2
SVM_test_pred.1$score3 <-
SVM_test_pred.1$SVM1_rel3+SVM_test_pred.1$SVM2_rel3+SVM_test_pred.1$SVM
3_rel3
SVM_test_pred.1$score5 <-
SVM_test_pred.1$SVM1_rel5+SVM_test_pred.1$SVM2_rel5+SVM_test_pred.1$SVM
3_rel5

SVM_test_pred.1$sum <-
SVM_test_pred.1$score2+SVM_test_pred.1$score3+SVM_test_pred.1$score5

SVM_test_pred.1$pred.CL <- ifelse(SVM_test_pred.1$score2 >
SVM_test_pred.1$score3 & SVM_test_pred.1$score2 >
SVM_test_pred.1$score5,"CL2",
                                  ifelse(SVM_test_pred.1$score3 >
SVM_test_pred.1$score2 & SVM_test_pred.1$score3 >
SVM_test_pred.1$score5,"CL3" ,
                                         ifelse(SVM_test_pred.1$score5 >
SVM_test_pred.1$score2 & SVM_test_pred.1$score5 >
SVM_test_pred.1$score3,"CL5", NA)))
```

```
SVM_test_pred.1$reliability <- ifelse(SVM_test_pred.1$pred.CL == "CL2", SVM_test_pred.1$score2/SVM_test_pred.1$sum,
                                      ifelse(SVM_test_pred.1$pred.CL == "CL3", SVM_test_pred.1$score3/SVM_test_pred.1$sum,

ifelse(SVM_test_pred.1$pred.CL == "CL5", SVM_test_pred.1$score5/SVM_test_pred.1$sum,NA)))

SVM_test_pred.1$true.CL <- test_labels

tail(SVM_test_pred.1)
table(predict=SVM_test_pred.1$pred.CL, real=SVM_test_pred.1$true.CL)
```