accelerometer-svm-proj7

November 28, 2017

0.0.1 1. Processing data

In [5]:
```
'''
#Setting up SVM; processing input data in right format as input to svm.
'''

from sklearn import svm
from sklearn import svm, datasets
from sklearn.model_selection import GridSearchCV

import numpy as np

d00 = np.genfromtxt('sleeping_features.csv',delimiter=',')
d01 = np.genfromtxt('sitting_features.csv',delimiter=',')
d02 = np.genfromtxt('standing_features.csv',delimiter=',')
d03 = np.genfromtxt('walking_features.csv',delimiter=',')


n00 = d00.shape[0]
n01 = d01.shape[0]
n02 = d02.shape[0]
n03 = d03.shape[0]

#print(n00)
#print(n01)
#print(n02)
#print(n03)

t00 = np.zeros(n00) + 1
t01 = np.zeros(n01) + 2
t02 = np.zeros(n02) + 3
t03 = np.zeros(n03) + 4


d10 = np.vstack((d00,d01))
d10 = np.vstack((d10,d02))
d10 = np.vstack((d10,d03))
```

```
        #print(d10)
        print(d10.shape)

        t10 = np.concatenate((t00,t01))
        t10 = np.concatenate((t10,t02))
        t10 = np.concatenate((t10,t03))

        #print(t10)
        print(t10.shape)

        from sklearn.utils import shuffle
        d11, t11 = shuffle(d10, t10, random_state=0)

        #print(d11,t11)
```

```
(2347, 8)
(2347,)
```

0.0.2 2. Cross validation.

0.0.3 Feature set: meanX, meanY, meanZ, varX, varY, varZ, meanMax, meanMin.

0.0.4 'C'=1, 'gamma'=auto

```
In [6]: #cross validation over 10 differnt combinations of the data set

        from sklearn.model_selection import cross_val_score

        clf = svm.SVC()
        print(clf)

        scores = cross_val_score(clf, d11, t11, cv=10)
        print(scores)
        print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
```

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
[ 1.  1.  1.  1.  1.  1.  1.  1.  1.  1.]
Accuracy: 1.00 (+/- 0.00)
```

Cross validation show that the model fitted with paramaeters 'C'=1 and 'gamma'=auto (default parameters) results in a consistently good accuracy model.

0.0.5 3. SVM model fit.

In [8]:
```
#SVM classifier.  Feature set: X, y, z values
from sklearn import svm
from sklearn import svm, datasets
from sklearn.model_selection import GridSearchCV

import numpy as np


dt01 = np.genfromtxt('sitting_short_features.csv', delimiter=',')
dt02 = np.genfromtxt('standing_short_features.csv', delimiter=',')
dt03 = np.genfromtxt('standing_short_features1.csv', delimiter=',')

clf2 = svm.SVC()
print(clf2)
clf2.fit(d11,t11)
zz01 = clf2.predict(dt01)
nzz01 = zz01.shape[0]
#print(zz01)
ac01 = (zz01 == 1.).sum()
print(ac01, nzz01)
ac011 = (ac01/nzz01)*100
print("accuracy: ", ac011)
print("error: ", 100-ac011)

zz02 = clf2.predict(dt02)
nzz02 = zz02.shape[0]
#print(zz02)
ac02 = (zz02 == 3.).sum()
print(ac02, nzz02)
ac021 = (ac02/nzz02)*100
print("accuracy: ", ac021)
print("error: ", 100-ac021)

zz03 = clf2.predict(dt03)
nzz03 = zz03.shape[0]
#print(zz03)
ac03 = (zz03 == 3.).sum()
print(ac03, nzz03)
ac031 = (ac03/nzz03)*100
print("accuracy: ", ac031)
print("error: ", 100-ac031)
```

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
115 115
```

```
accuracy:  100.0
error:  0.0
116 116
accuracy:  100.0
error:  0.0
116 119
accuracy:  97.4789915966
error:  2.52100840336
```

The SVM model fit with x,y,z values as feature set results prediction error rate 0.08(%)

```
In [ ]:
```