accelerometer-svm

November 13, 2017

0.0.1 1. Processing data

In [12]:
```
'''
#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

#--------------------------------

def processData(indata):
    sdata = np.hsplit(indata, np.array([1,4,5,8,9,12,13,16,17,20]))

    timestamps = sdata[0]
#    print(timestamps)

    odata1 = sdata[1]
    otdata = sdata[2]
    odata2 = (odata1*odata1).sum(axis=1)**0.5
    for i in range(1,5):
#        print(i)
        odata1 = np.vstack((odata1,sdata[2*i+1]))
        otdata = np.vstack((otdata,sdata[2*i+2]))
        dd = (sdata[2*i+1]*sdata[2*i+1]).sum(axis=1)**0.5
        odata2 = np.concatenate((odata2,dd))

    otdata = np.ravel(otdata)
    odata2 = odata2.reshape(-1,1)

    return odata1, odata2, otdata, timestamps

#--------------------------------

indata = np.genfromtxt('indata2.csv',skip_header=1,delimiter=',')
```

```
#print(indata)
#print(indata.shape)

n_data = indata.shape[0]

n_trdata = (n_data*1)//3
ttdata = indata[:n_trdata]
trdata = indata[n_trdata:]

#training data
trdata1,trdata2,trtargetdata, ts = processData(trdata)

#test data
ttdata1,ttdata2,tttargetdata, ts2 = processData(ttdata)
```

0.0.2 2. SVM model fit. Feature set x, y, z values.

In [13]:
```
#SVM classifier.  Feature set: X, y, z values

classifier1 = svm.SVC(C=1, gamma=0.6)
print(classifier1)
classifier1.fit(trdata1,trtargetdata)

scoreval = classifier1.score(ttdata1,tttargetdata)
print("score:", scoreval)
print('accuracy:', scoreval*100, '%')
```

```
SVC(C=1, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.6, kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
score: 0.9472
accuracy: 94.72 %
```

The SVM model fit with x,y,z values as feature set results in a good fit with accuracy around 94.72%. This result and accuracy of the model is further confirmed using cross validation over 10 combinations of the data set.

0.0.3 3. SVM model fit. Magnitude of acceleration as feature set

In [14]:
```
#SvM classifier. Feature set: magnitude of the acceration vector (sqrt(x*x+y*y+z*z))

clf2 = svm.SVC(C=1, gamma=0.6)
print(clf2)
clf2.fit(trdata2,trtargetdata)
scoreval = clf2.score(ttdata2,tttargetdata)
print("score:", scoreval)
print('accuracy:', scoreval*100, '%')
```

```
SVC(C=1, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.6, kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
score: 0.530133333333
accuracy: 53.0133333333 %
```

SVM model fit with magnitude as feature set is not a very good fit. It results in accuracy of around 53% and is discarded for later use.

0.0.4 4. GridSearchCV to search for best parameters 'C' and 'gamma'

```
In [15]: #using GridSearchCV to check for the best parameter values.

         param_grid = [
           {'C': [1, 10, 100, 1000], 'gamma': [0.3, 0.333, 0.4, 0.5, 0.6, 0.7], 'kernel': ['rbf'
          ]

         svc = svm.SVC()
         clf = GridSearchCV(svc, param_grid)
         print(clf)
         clf.fit(trdata1, trtargetdata)
         print(sorted(clf.cv_results_.keys()))

         scoreval = clf.score(ttdata1,tttargetdata)
         print("score:", scoreval)
         print('accuracy:', scoreval*100, '%')
```

```
GridSearchCV(cv=None, error_score='raise',
       estimator=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),
       fit_params=None, iid=True, n_jobs=1,
       param_grid=[{'C': [1, 10, 100, 1000], 'gamma': [0.3, 0.333, 0.4, 0.5, 0.6, 0.7], 'kernel'
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
       scoring=None, verbose=0)
['mean_fit_time', 'mean_score_time', 'mean_test_score', 'mean_train_score', 'param_C', 'param_ga
score: 0.946266666667
accuracy: 94.6266666667 %
```

```
In [18]: clf.get_params()
```

```
Out[18]: {'C': 1.0,
          'cache_size': 200,
          'class_weight': None,
          'coef0': 0.0,
```

```
 'decision_function_shape': 'ovr',
 'degree': 3,
 'gamma': 0.6,
 'kernel': 'rbf',
 'max_iter': -1,
 'probability': False,
 'random_state': None,
 'shrinking': True,
 'tol': 0.001,
 'verbose': False}
```

GridSearchCV with different combinations of the parameters 'C' and 'gamma' results in the best fit SVM model for values 'C' = 1 and 'gamma' = 0.6.

0.0.5 5. Cross validation. Feature set: x, y, z values. 'C'=1, 'gamma'=0.6

In [19]:
```
#cross validation over 10 differnt combinations of the data set

from sklearn.model_selection import cross_val_score

clf = svm.SVC(gamma=0.6)
print(clf)

scores = cross_val_score(clf, trdata1, trtargetdata, 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=0.6, kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
[ 0.93733333  0.946       0.91666667  0.924       0.93666667  0.90666667
  0.942       0.934       0.93533333  0.92933333]
Accuracy: 0.93 (+/- 0.02)
```

Cross validation show that the model fitted with paramaeters 'C'=1 and 'gamma'=0.6 results in a consistently good accuracy model.