3train3testLR

December 13, 2017

In [17]:
```
import pandas as pd
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_auc_score
import numpy as np
from pandas import DataFrame,Series
from matplotlib.colors import ListedColormap
from random import sample
import matplotlib.pyplot as plt
from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import cross_val_score
```

In [ ]:
```
#Description of features
#Average[3]: Average acceleration (for each axis)
#Standard Deviation[3]: Standard deviation (for each axis)
#Average Absolute Difference[3]: Average absolute
#difference between the value of each of the 200 readings
#within the ED and the mean value over those 200 values
#(for each axis)
#Average Resultant Acceleration[1]: Average of the square
#roots of the sum of the values of each axis squared
#over the ED
#Time Between Peaks[3]: Time in milliseconds between
#peaks in the sinusoidal waves associated with most
#activities (for each axis)
#Binned Distribution[30]: We determine the range of values
#for each axis (maximum  minimum), divide this range into
#10 equal sized bins, and then record what fraction of the
#200 values fell within each of the bins.
```

In [20]:
```
#Importing the dataset
df_features = pd.read_csv("H:/mastersProject/activity_analyzer/LogisticRegression/Data
X_data = df_features.values[:, 2:45]
y_data = df_features.values[:, 45]
print(len(df_features))
```

417

In [25]:
```
# Data 3 people for training the model
lr = LogisticRegression(C=100.0, random_state=1)
X_train, X_test, y_train, y_test = train_test_split(X_data, y_data, test_size = 0.30,
probas = lr.fit(X_train, y_train)
predict = lr.predict(X_test)
logisticRegScore = lr.score(X_test, y_test)
print(lr.coef_)
print("Score of Logistic regression=", logisticRegScore)
```

```
[[ -4.26355078e-02  -1.08800655e-01  -1.18109762e-01  -6.98166698e-02
   -1.26457660e-01  -4.19031457e-02   1.56527014e-01   6.83777586e-02
   -1.59095593e-01  -7.86240681e-02  -5.60574777e-02  -3.31296621e-01
   -4.65490429e-01  -4.03954530e-01  -1.10690882e-01  -1.79563681e-01
    2.70517854e-01   3.51522240e-01   2.14428600e-01   7.35074257e-02
   -3.70309035e-02   1.59810737e-02   3.08821037e-01   5.74312214e-02
   -1.60570347e-03  -2.36263254e-01  -3.11438642e-01  -6.36148683e-02
   -6.77054416e-02  -1.33159005e-01   6.39061136e-03   4.96286921e-01
    3.70190201e-01  -3.91525716e+00  -3.18801967e+00  -1.33183169e+00
    1.00325076e+00   1.57433713e+00  -3.93544302e+00  -4.33108295e+00
   -3.60672448e+00  -1.10803655e+00   2.73663678e-01]]
Score of Logistic regression= 1.0
```

In [26]:
```
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, predict, labels=["walking", "running"])
print(cm)
```

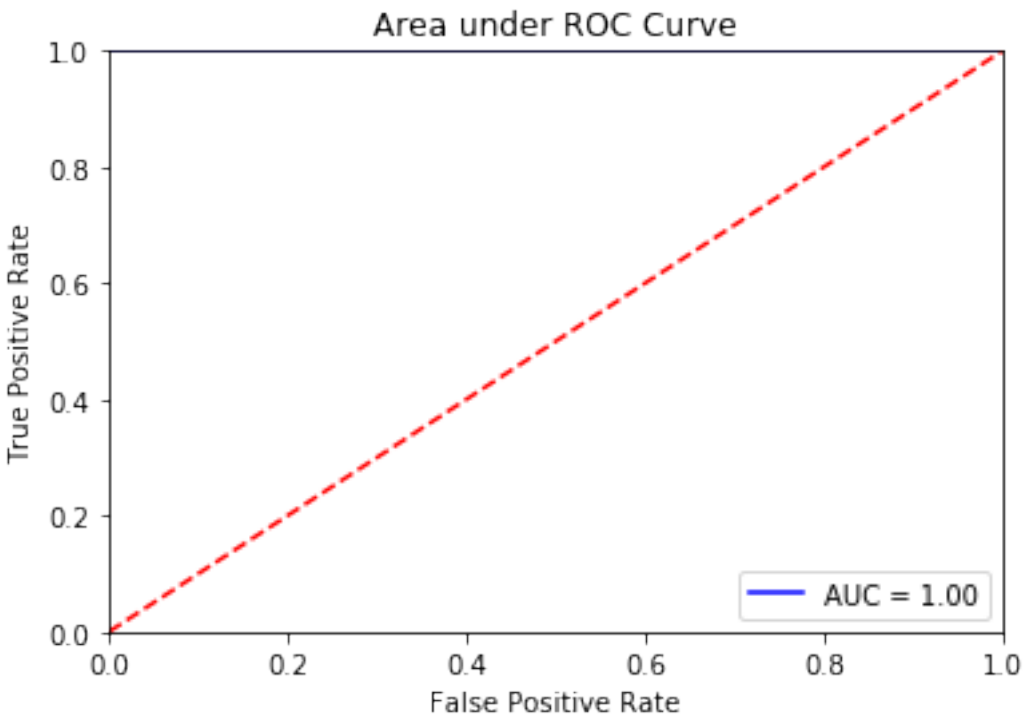
```
[[77  0]
 [ 0 49]]
```

In [14]:
```
# Cross validation using KFold
lr_data = pd.read_csv('H:/mastersProject/activity_analyzer/LogisticRegression/Data/fea
X = lr_data.values[:, 2:45]
y = lr_data.values[:, 45]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
lr = LogisticRegression(C=100.0, random_state=1)
lr.fit(X_train, y_train)
kfold = StratifiedKFold(n_splits=32, random_state=1).split(X_train, y_train)
scores = cross_val_score(estimator=lr, X=X_train, y=y_train, cv=10)
print(scores)
```

```
[ 1.          1.          1.          1.          1.          1.          1.
  1.          1.          0.96551724]
```

In [ ]:
```
#This model is overfitting as it has just 3 people data.
```

In [32]:
```
from sklearn.metrics import roc_curve
from sklearn.metrics import auc
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt
#Area under ROC
num_predns = []

for i in range(0,len(predict)):
    if predict[i] == "walking":
        num_predns.append(0)
    else:
        num_predns.append(1)

num_labels = []

for i in range(0,len(y_test)):
    if y_test[i] == "walking":
        num_labels.append(0)
    else:
        num_labels.append(1)

predns = np.array(num_predns)
labels = np.array(num_labels)

fpr, tpr, thresholds = roc_curve(labels, predns)
roc_auc = auc(fpr, tpr)
import matplotlib.pyplot as plt
plt.title('Area under ROC Curve')
plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```


Area under ROC Curve
True Positive Rate
False Positive Rate
AUC = 1.00
0.0
0.2
0.4
0.6
0.8
1.0