import numpy as np  
import pandas as pd   
import seaborn as sns   
import matplotlib.pyplot as plt  
%matplotlib notebook

from sklearn.model\_selection import train\_test\_split   
from sklearn.neighbors import KNeighborsClassifier  
from sklearn import svm   
from sklearn.metrics import classification\_report  
from sklearn.metrics import accuracy\_score, plot\_confusion\_matrix  
from sklearn.model\_selection import cross\_val\_score  
from sklearn.model\_selection import cross\_val\_predict  
from sklearn.svm import SVC  
from sklearn.feature\_extraction.text import CountVectorizer  
from sklearn.naive\_bayes import GaussianNB  
from sklearn.preprocessing import LabelEncoder

classes=[]  
for i in range(1,20):  
 classes.append("A"+str(i))

data = pd.read\_table('dailysportsdata.txt', delimiter=',', header=0)

data.head()

Activity

T-x\_acc (1)

T-y\_acc (1)

T-z\_acc (1)

T-x\_gryo (1)

T-y\_gryo (1)

T-z\_gryo (1)

T-x\_mag (1)

T-y\_mag (1)

T-z\_mag (1)

...

RL-z\_mag (125)

LL-x\_acc (125)

LL-y\_acc (125)

LL-z\_acc (125)

LL-x\_gryo (125)

LL-y\_gryo (125)

LL-z\_gryo (125)

LL-x\_mag (125)

LL-y\_mag (125)

LL-z\_mag (125)

0

A1

8.1305

1.0349

5.4217

-0.009461

0.001915

-0.003424

-0.78712

-0.069654

0.15730

...

-0.036874

-2.8154

-9.0600

2.6025

-0.003904

-0.006729

-0.009789

0.73897

0.30275

-0.056262

1

A1

7.9665

1.1684

5.6755

-0.005730

0.026995

-0.009029

-0.79062

-0.071635

0.13429

...

-0.038551

-2.8233

-9.0757

2.6337

-0.006769

-0.006575

-0.004326

0.74027

0.30192

-0.057155

2

A1

7.8917

1.1390

5.6980

0.014180

0.028722

-0.009079

-0.79531

-0.069460

0.12447

...

-0.040145

-2.8091

-9.0846

2.6295

-0.000714

-0.002681

0.004770

0.74072

0.30101

-0.057301

3

A1

7.9366

1.1536

5.6318

0.003242

0.029965

0.009111

-0.79292

-0.070358

0.13194

...

-0.041109

-2.8844

-9.0849

2.6298

-0.010604

-0.002827

-0.004194

0.74150

0.30305

-0.055743

4

A1

7.8913

1.1972

5.9082

-0.044333

-0.067467

-0.004235

-0.79592

-0.073174

0.12086

...

-0.039495

-2.8249

-9.1083

2.6322

0.013583

0.013670

0.007613

0.74007

0.30324

-0.055548

5 rows × 5626 columns

data.info

<bound method DataFrame.info of Activity T-x\_acc (1) T-y\_acc (1) T-z\_acc (1) T-x\_gryo (1) \  
0 A1 8.1305 1.03490 5.42170 -0.009461   
1 A1 7.9665 1.16840 5.67550 -0.005730   
2 A1 7.8917 1.13900 5.69800 0.014180   
3 A1 7.9366 1.15360 5.63180 0.003242   
4 A1 7.8913 1.19720 5.90820 -0.044333   
... ... ... ... ... ...   
9115 A19 7.2364 -1.88630 -0.63428 -0.554970   
9116 A19 11.3240 0.36680 2.03820 -0.011767   
9117 A19 10.1070 0.43644 0.02232 -2.356700   
9118 A19 14.0750 -2.00440 0.29813 1.908900   
9119 A19 8.3123 0.96283 3.48250 -0.770440   
  
 T-y\_gryo (1) T-z\_gryo (1) T-x\_mag (1) T-y\_mag (1) T-z\_mag (1) ... \  
0 0.001915 -0.003424 -0.78712 -0.069654 0.15730 ...   
1 0.026995 -0.009029 -0.79062 -0.071635 0.13429 ...   
2 0.028722 -0.009079 -0.79531 -0.069460 0.12447 ...   
3 0.029965 0.009111 -0.79292 -0.070358 0.13194 ...   
4 -0.067467 -0.004235 -0.79592 -0.073174 0.12086 ...   
... ... ... ... ... ... ...   
9115 -0.324660 0.712660 -0.77909 -0.464200 -0.11567 ...   
9116 0.409540 0.169110 -0.75026 -0.228890 -0.47219 ...   
9117 0.076880 -0.352710 -0.71162 0.451740 -0.24060 ...   
9118 -0.729320 0.397400 -0.77983 -0.242090 -0.37249 ...   
9119 0.273500 -0.561200 -0.61446 -0.547290 -0.32384 ...   
  
 RL-z\_mag (125) LL-x\_acc (125) LL-y\_acc (125) LL-z\_acc (125) \  
0 -0.036874 -2.8154 -9.06000 2.6025   
1 -0.038551 -2.8233 -9.07570 2.6337   
2 -0.040145 -2.8091 -9.08460 2.6295   
3 -0.041109 -2.8844 -9.08490 2.6298   
4 -0.039495 -2.8249 -9.10830 2.6322   
... ... ... ... ...   
9115 0.230040 -14.7580 -0.34863 -4.2297   
9116 0.434830 -8.7282 3.59850 -2.6440   
9117 -0.298710 -16.5650 2.46170 -2.4732   
9118 -0.090748 -13.1390 -0.61176 6.5424   
9119 -0.212350 3.6284 15.05400 7.0935   
  
 LL-x\_gryo (125) LL-y\_gryo (125) LL-z\_gryo (125) LL-x\_mag (125) \  
0 -0.003904 -0.006729 -0.009789 0.73897   
1 -0.006769 -0.006575 -0.004326 0.74027   
2 -0.000714 -0.002681 0.004770 0.74072   
3 -0.010604 -0.002827 -0.004194 0.74150   
4 0.013583 0.013670 0.007613 0.74007   
... ... ... ... ...   
9115 -1.058200 0.655410 0.636460 0.53204   
9116 1.360200 0.113690 0.127500 0.73818   
9117 -1.095000 -1.234900 0.604770 0.75904   
9118 1.248700 -0.016356 0.546180 0.73540   
9119 -3.677800 0.151150 -0.740350 0.47028   
  
 LL-y\_mag (125) LL-z\_mag (125)   
0 0.30275 -0.056262   
1 0.30192 -0.057155   
2 0.30101 -0.057301   
3 0.30305 -0.055743   
4 0.30324 -0.055548   
... ... ...   
9115 0.75974 0.228250   
9116 0.40739 -0.250950   
9117 0.52748 0.072323   
9118 0.27723 0.448810   
9119 0.43618 0.545110   
  
[9120 rows x 5626 columns]>

CLASS = data.Activity   
CLASS.head()  
list = ['Activity']  
X = data.drop(list,axis = 1 )

data1 = pd.read\_table('dailysportsdataPCA1.txt', delimiter=',', header=0)

data1.head()

Attribute 1

Attribute 2

Attribute 3

Attribute 4

Attribute 5

Attribute 6

Attribute 7

Attribute 8

Attribute 9

Attribute 10

...

Attribute 36

Attribute 37

Attribute 38

Attribute 39

Attribute 40

Attribute 41

Attribute 42

Attribute 43

Attribute 44

Attribute 45

0

-1.637071

-0.838735

3.347988

-9.458428

-7.743542

-23.807102

-9.581168

-7.642276

-11.361543

-0.266466

...

2.073364

0.974592

0.791543

-0.700301

-0.404438

-0.751582

-0.741852

-0.034034

0.985440

0.534058

1

-1.749408

-0.970375

3.377247

-9.282271

-7.725263

-23.857871

-9.762795

-7.540563

-11.318303

-0.268322

...

1.984538

0.980023

1.040050

-0.856222

-0.342692

-0.634753

-0.762791

0.138708

0.865257

0.542783

2

-1.748035

-1.016446

3.374029

-9.110532

-7.734951

-23.876501

-9.873329

-7.561801

-11.326528

-0.282362

...

2.000784

1.218666

0.824946

-0.764731

-0.326847

-0.702696

-0.805013

0.099017

0.865078

0.519723

3

-1.700473

-1.116532

3.374351

-9.011856

-7.627430

-23.929857

-9.868562

-7.549670

-11.321522

-0.296492

...

2.038874

1.079182

0.837559

-0.729533

-0.353082

-0.731957

-0.783896

0.075949

0.857791

0.500859

4

-1.590936

-1.198873

3.333404

-8.908969

-7.557411

-23.990856

-9.868686

-7.565691

-11.325635

-0.277031

...

2.000701

1.081808

0.925057

-0.786908

-0.320531

-0.655008

-0.781665

0.067867

0.879226

0.516790

5 rows × 45 columns

data1.info

<bound method DataFrame.info of Attribute 1 Attribute 2 Attribute 3 Attribute 4 Attribute 5 \  
0 -1.637071 -0.838735 3.347988 -9.458428 -7.743542   
1 -1.749408 -0.970375 3.377247 -9.282271 -7.725263   
2 -1.748035 -1.016446 3.374029 -9.110532 -7.734951   
3 -1.700473 -1.116532 3.374351 -9.011856 -7.627430   
4 -1.590936 -1.198873 3.333404 -8.908969 -7.557411   
... ... ... ... ... ...   
9115 -14.293856 -14.116724 0.828153 -0.751334 0.542264   
9116 -2.732382 -17.674730 -5.787976 7.303718 4.846983   
9117 -6.944206 6.739651 -26.469235 2.396547 -9.336720   
9118 -18.624201 -22.287414 13.653796 8.434307 -8.786071   
9119 -14.740642 -17.646131 12.134152 12.154559 -7.454618   
  
 Attribute 6 Attribute 7 Attribute 8 Attribute 9 Attribute 10 ... \  
0 -23.807102 -9.581168 -7.642276 -11.361543 -0.266466 ...   
1 -23.857871 -9.762795 -7.540563 -11.318303 -0.268322 ...   
2 -23.876501 -9.873329 -7.561801 -11.326528 -0.282362 ...   
3 -23.929857 -9.868562 -7.549670 -11.321522 -0.296492 ...   
4 -23.990856 -9.868686 -7.565691 -11.325635 -0.277031 ...   
... ... ... ... ... ... ...   
9115 15.952670 -0.965580 4.486552 -11.194345 3.239676 ...   
9116 10.606923 3.092321 2.177551 -2.438917 6.407283 ...   
9117 -2.572455 10.498769 30.389892 -14.748274 6.116827 ...   
9118 4.507573 8.573041 -4.581754 -3.421447 -3.207566 ...   
9119 6.225593 10.534479 -7.282178 -2.400744 -10.717747 ...   
  
 Attribute 36 Attribute 37 Attribute 38 Attribute 39 Attribute 40 \  
0 2.073364 0.974592 0.791543 -0.700301 -0.404438   
1 1.984538 0.980023 1.040050 -0.856222 -0.342692   
2 2.000784 1.218666 0.824946 -0.764731 -0.326847   
3 2.038874 1.079182 0.837559 -0.729533 -0.353082   
4 2.000701 1.081808 0.925057 -0.786908 -0.320531   
... ... ... ... ... ...   
9115 9.203627 14.396985 -9.561305 10.104632 -15.151837   
9116 11.361846 2.968851 -9.171966 7.517901 -4.520726   
9117 1.598336 -7.671363 -7.473155 16.088170 18.020874   
9118 -2.551254 -9.069357 16.391776 -21.867964 -0.773144   
9119 -0.260988 -4.717039 13.047815 -17.054062 -4.162224   
  
 Attribute 41 Attribute 42 Attribute 43 Attribute 44 Attribute 45   
0 -0.751582 -0.741852 -0.034034 0.985440 0.534058   
1 -0.634753 -0.762791 0.138708 0.865257 0.542783   
2 -0.702696 -0.805013 0.099017 0.865078 0.519723   
3 -0.731957 -0.783896 0.075949 0.857791 0.500859   
4 -0.655008 -0.781665 0.067867 0.879226 0.516790   
... ... ... ... ... ...   
9115 -9.684862 9.899371 -1.396864 -5.095063 -10.930188   
9116 0.763019 -5.394976 0.142910 -0.913324 3.723007   
9117 -14.690267 -7.987767 2.643529 10.971603 2.095462   
9118 13.575252 -7.642678 0.263808 -2.206267 -1.478670   
9119 13.097214 -7.291438 5.170682 -7.891294 -8.677844   
  
[9120 rows x 45 columns]>

x\_train, x\_test, y\_train, y\_test = train\_test\_split(data1, CLASS, test\_size=0.25, random\_state=100)

# K-Nearest Neighbour

KNN = KNeighborsClassifier(n\_neighbors=6,weights='distance')

KNN

KNeighborsClassifier(n\_neighbors=6, weights='distance')

KNN.fit(x\_train, y\_train)

KNeighborsClassifier(n\_neighbors=6, weights='distance')

y\_pred = KNN.predict(x\_test)

print(classification\_report(y\_test, y\_pred))

precision recall f1-score support  
  
 A1 1.00 1.00 1.00 126  
 A10 0.97 0.98 0.98 132  
 A11 0.97 0.97 0.97 120  
 A12 1.00 1.00 1.00 127  
 A13 0.90 1.00 0.95 108  
 A14 1.00 1.00 1.00 117  
 A15 1.00 1.00 1.00 123  
 A16 1.00 0.98 0.99 128  
 A17 1.00 1.00 1.00 128  
 A18 1.00 0.97 0.98 120  
 A19 1.00 0.62 0.77 119  
 A2 0.94 1.00 0.97 136  
 A3 1.00 1.00 1.00 114  
 A4 1.00 1.00 1.00 112  
 A5 0.93 1.00 0.96 110  
 A6 0.95 1.00 0.97 116  
 A7 0.74 1.00 0.85 106  
 A8 0.87 0.70 0.78 119  
 A9 0.98 1.00 0.99 119  
  
 accuracy 0.96 2280  
 macro avg 0.96 0.96 0.96 2280  
weighted avg 0.96 0.96 0.96 2280

model = KNeighborsClassifier(n\_neighbors=6,weights='distance')

scores=cross\_val\_score(model, x\_train, y\_train, cv=10)  
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() \* 2))

Accuracy: 0.96 (+/- 0.01)

y\_pred = cross\_val\_predict(model,x\_train,y\_train,cv=10)

y\_pred

array(['A6', 'A18', 'A11', ..., 'A17', 'A15', 'A12'], dtype=object)

# Support Vector Machine

model = SVC(random\_state=42, kernel='linear')

scores=cross\_val\_score(model, x\_train, y\_train, cv=10)  
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() \* 2))

Accuracy: 0.97 (+/- 0.01)

clf\_svm = SVC(random\_state=42, kernel='rbf')  
clf\_svm.fit(x\_train,y\_train)

SVC(random\_state=42)

plot\_confusion\_matrix(clf\_svm,  
 x\_test,  
 y\_test,  
 values\_format = 'd')  
plt.show()

<IPython.core.display.Javascript object>

# Naive Bayes

data1.shape

(9120, 45)

x\_train, x\_test, y\_train, y\_test = train\_test\_split(data1, CLASS, test\_size=0.35, random\_state=100)

x\_test[:10]

Attribute 1

Attribute 2

Attribute 3

Attribute 4

Attribute 5

Attribute 6

Attribute 7

Attribute 8

Attribute 9

Attribute 10

...

Attribute 36

Attribute 37

Attribute 38

Attribute 39

Attribute 40

Attribute 41

Attribute 42

Attribute 43

Attribute 44

Attribute 45

3557

-13.816802

-7.531889

27.902203

9.848131

3.101119

9.469981

1.165715

3.839769

5.433657

0.109294

...

-0.090079

1.421122

-0.599632

-0.446465

0.548195

1.068415

0.402151

0.185730

0.539437

-0.769784

5195

10.614481

-12.930692

-26.130464

9.120812

-5.275665

-0.923854

4.485880

1.395949

-0.376214

4.100316

...

0.702761

-0.532732

-1.380617

-1.674248

-0.722167

-0.073333

-3.362886

7.824435

-0.072566

-4.080893

942

-33.954150

10.389130

-1.674615

-7.566204

-0.994653

-1.023016

0.873867

6.467057

8.828426

0.010703

...

-0.857158

-0.743140

-0.461022

0.599348

0.124897

-0.341702

0.320171

-0.455542

-0.279942

0.498503

6295

11.277405

-18.766921

-11.232132

9.613665

-10.852778

-4.077367

9.477659

-1.276162

2.933427

0.857485

...

-1.442706

-2.292571

-0.593499

-0.821423

0.252108

2.532633

-3.123379

4.463212

-3.492100

-5.114868

7314

18.308248

1.729160

-13.683737

-24.091873

-29.306101

-13.485991

-5.662506

1.446476

1.102319

-2.514169

...

-2.150370

0.556690

0.094956

2.609883

-0.671382

1.362829

1.086044

-1.519743

3.573919

3.578553

2513

-32.844590

-9.131236

27.056085

-6.465059

4.867199

1.210376

7.298532

-2.922950

-2.906895

2.240128

...

0.298875

0.353525

1.862457

-1.638984

0.170207

-0.080248

-0.639816

-1.544993

-0.656108

-0.189646

6100

15.914793

-19.581466

-2.403929

7.264876

-6.739648

12.642107

22.221420

-3.776180

-5.052369

0.175612

...

-0.007842

0.495968

-0.323441

-1.239404

3.677457

1.588052

-1.642665

-2.601911

-5.174721

0.314754

3267

-37.926811

1.114966

24.926525

-2.849008

0.931189

-1.617049

4.346599

2.779779

4.451548

-0.036642

...

0.098846

0.614946

-0.283311

-0.465992

0.946345

0.567126

0.129178

1.184687

1.186429

-0.842729

3610

-17.801781

3.837606

-4.678446

0.578484

9.387558

2.852255

3.589843

3.035833

5.300868

-0.509429

...

0.531912

0.717635

0.086139

-0.344420

0.195382

-0.052331

0.035352

-0.001315

-0.685395

-1.001539

2365

-10.518993

18.735219

-12.775085

-13.854505

5.721514

9.384427

-7.763612

9.794939

10.517735

-1.005409

...

0.256791

1.181348

0.116649

-2.495403

0.088026

-0.198556

1.458225

1.591004

4.342062

-0.667352

10 rows × 45 columns

y\_test[:10]

3557 A8  
5195 A11  
942 A2  
6295 A14  
7314 A16  
2513 A6  
6100 A13  
3267 A7  
3610 A8  
2365 A5  
Name: Activity, dtype: object

NBmodel = GaussianNB()  
NBmodel.fit(x\_train, y\_train)

GaussianNB()

y\_predicted = NBmodel.predict(x\_test)

y\_predicted

array(['A7', 'A11', 'A2', ..., 'A16', 'A15', 'A7'], dtype='<U3')

accuracy\_score(y\_test, y\_predicted)\*100

93.23308270676691

print(classification\_report(y\_test,y\_predicted))

precision recall f1-score support  
  
 A1 0.99 0.99 0.99 171  
 A10 0.75 0.80 0.78 168  
 A11 0.69 0.70 0.70 178  
 A12 0.98 1.00 0.99 177  
 A13 0.96 1.00 0.98 158  
 A14 1.00 0.98 0.99 153  
 A15 1.00 1.00 1.00 187  
 A16 1.00 0.94 0.97 179  
 A17 1.00 1.00 1.00 170  
 A18 0.99 0.97 0.98 159  
 A19 0.89 0.94 0.91 167  
 A2 0.94 0.99 0.97 187  
 A3 0.99 0.96 0.98 170  
 A4 0.97 1.00 0.98 160  
 A5 0.99 1.00 1.00 155  
 A6 0.96 0.98 0.97 165  
 A7 0.82 0.93 0.87 151  
 A8 0.95 0.66 0.78 167  
 A9 0.87 0.87 0.87 170  
  
 accuracy 0.93 3192  
 macro avg 0.93 0.93 0.93 3192  
weighted avg 0.93 0.93 0.93 3192