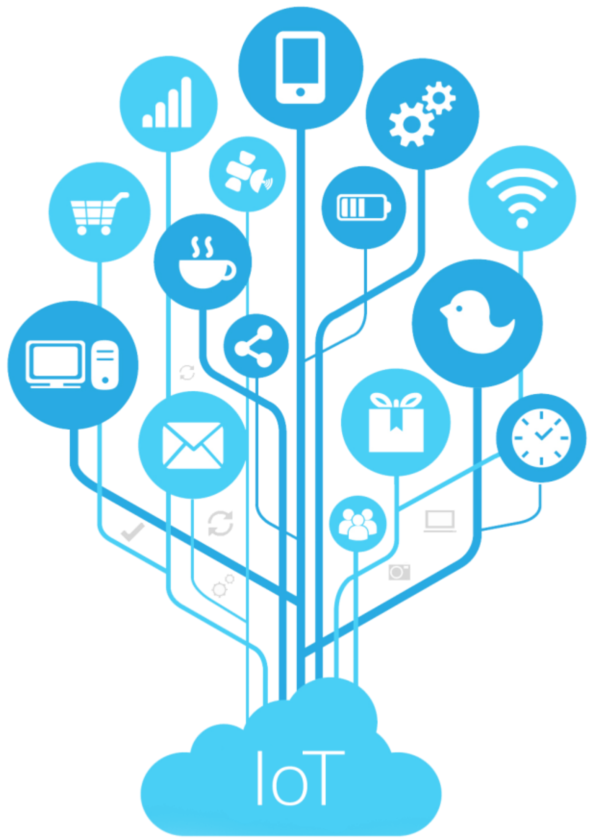

Activities of Daily Living Analysis

Venkata Sai Praneeth Kandula

Data:

The dataset used for this analysis contains tri-axial accelerometer readings while performing a number of simple human activities which are defined as Human Motion Primitives (HMP). In this analysis we will only be looking at three of these primitives:

1. **climb_stairs**: to climb a number of steps of a staircase.
2. **drink_glass**: to pick a glass from a table, drink and put it back on the table.
3. **getup_bed**: to get up from a lying position on a bed.

Each file in the dataset corresponds to a volunteer performing an activity for a specific time period. For example, the file **Accelerometer-2011-03-24-10-24-39-climb_stairs-f1.txt** refers to an accelerometer recording that was taken on March 24, 2011, starting from 10:24.39 a.m. The recording refers to the HMP "climb_stairs" executed by the volunteer with ID "f1".

Each raw file consists of three-tab separated columns of data, each corresponding to acceleration readings from an accelerometer attached to the right wrist of the user along three axes:

1. x axis: pointing toward the hand
2. y axis: pointing toward the left
3. z axis: perpendicular to the plane of the hand

There were a total of 303 files, 102 corresponding to climb_stairs, 100 corresponding to drink_glass and 101 corresponding to getup_bed.

To create the dataset, we read each of these files into R and then performed feature engineering to extract useful information that captures information that is characteristic of each of the activities, each file represents a single observation (row) in the resulting dataset which can be seen in the figure below.

Filter

	vol	HMPactivity	corxy	coryz	corxz	fftxmax	specenergy	varx	vary	varz	meanxsq
147	f1	Drink_glass	-0.774804879	-0.665463446	0.43854745	10506	227.4590	38.438727	47.971470	9.950625	1194.3390
295	m10	Getup_bed	-0.387373433	-0.675059994	0.69091784	16612	307.5289	125.948641	26.977911	44.881592	883.1210
260	m4	Getup_bed	-0.831417011	-0.448016008	0.55020883	16989	293.8128	151.342832	65.152904	22.590197	933.6542
300	m10	Getup_bed	0.226288083	0.381931353	0.81156399	10418	330.8037	164.356830	9.984225	56.165816	598.0665
59	m5	Climb_stairs	-0.067388086	0.135261774	-0.38107692	5361	244.9875	31.236499	23.875649	9.818358	193.8966
213	m1	Getup_bed	-0.258138139	-0.211370340	0.68558680	8344	311.6249	115.474548	34.428890	55.507661	758.3389
243	f4	Getup_bed	-0.225022430	-0.427241074	0.85751797	6984	207.9693	35.970500	52.492161	61.329312	327.3662
104	f1	Drink_glass	-0.849547243	-0.840907384	0.91487372	29962	253.4737	43.031851	70.093851	10.862912	1341.6369
259	m4	Getup_bed	-0.730584629	-0.571969791	0.64485330	19217	308.8184	152.069465	51.359963	21.411096	951.4700
85	m7	Climb_stairs	-0.457755172	0.072542828	0.06620877	3538	168.7282	29.825351	3.828058	3.126031	156.2942
273	f2	Getup_bed	-0.556992172	0.148435746	0.25653291	14465	308.8538	61.260407	91.496778	37.116078	481.7992
84	m7	Climb_stairs	-0.443494922	0.288646721	-0.08151356	3856	168.8703	27.840523	4.372699	3.008765	158.7929
66	m5	Climb_stairs	-0.338220726	0.212606239	0.25721155	6851	240.0418	20.264588	19.932079	12.422815	256.2243
43	f2	Climb_stairs	-0.157228170	0.541567104	0.13275262	10030	209.3408	8.574550	6.357319	5.274074	155.2423

Showing 1 to 14 of 303 entries

Figure 1 Head of the Dataset with Features

Exploratory Data Analysis:

Extracting feature information was a very challenging task as the raw data had just three variables of acceleration each along the x, y and z axis which was very hard to interpret. So we had to visualize the data graphically to compare, contrast and investigate underlying patterns in the data.

Below we can see the plots of the acceleration along the three axes corresponding to the three Human Primitives.

Figure 2 Tri-axial acceleration Graphs

We can see from the above graphs the difference in the changes of accelerations during each activity. We can see a clear distinguishing pattern in the x-axis acceleration amongst all the three activities.

During the drink_glass activity, it increases rapidly and then reaches a plateau during the drinking process and then comes down to its' normal level.

During the climb_stairs activity, it moves in the form of a sinusoidal waveform with each cycle representing the climbing of a single stair.

The accelerations along the y and z axis deviate in the middle during drink_glass but otherwise there is no such discernable pattern while performing the climb_stairs and getup_bed.

Feature Engineering:

1. corxy - Correlation between x and y
2. coryz - Correlation between x and y
3. corxz - Correlation between x and y
4. medianx - median of acceleration along x-axis
5. fftxmax - max(abs(Mod(fftx))), Peak Amplitude of the Fourier Transform of a signal. This is a characteristic of the macro behavior of the cyclic pattern of climbing the stairs.
6. specenergy - (sum(Mod(fftx))/length(fftx))+(sum(Mod(ffty))/length(ffty))+(sum(Mod(fftz))/length(fftz)), The total Spectral Energy of a signal.
7. Xbtwnzandy - mean((dat$z_acc > dat$x & dat$y_acc < dat$x_acc)) This is a Characteristic of the macro behavior of picking up glass and drinking. You can see from figure 2, the x_acc crosses over the y_acc and then falls back down after the activity is done.
8. meanxsq = mean(dat$x_acc)**2
9. varx – Variance along x-axis
10. vary – Variance along y-axis
11. varz – Variance along z-axis

Naïve Bayes Model:

Using the features obtained from above, we used a Bayesian classifier to predict the activity. The dataset in figure 1 had a total of 303 rows which was split into training and testing sets each containing 228 observations and 75 observations respectively.

Total Observations in Table: 75

Predictions	Actual Climb_stairs	Drink_glass	Getup_bed	Row Total
Climb_stairs	21 0.280	0 0.000	1 0.013	22
Drink_glass	0 0.000	31 0.413	1 0.013	32
Getup_bed	2 0.027	0 0.000	19 0.253	21
Column Total	23	31	21	75

Figure 3 Classification Matrix

We can see from figure 3 that we were able to predict correctly 71 observations out of a total of 75 observations which gives us an accuracy of 94.66%. Simulating the Naïve Bayes model 1000 times and taking the mean of the accuracies gives us a more precise value of 90.96%.

Principal Component Analysis

Performing principal component analysis on the dataset using the preprocess function from the caret package gives us 6 Principal components that capture 95% of the variance. The loadings of the Principal Components can be seen in figure 4 below:

	PC1	PC2	PC3	PC4	PC5	PC6
corxy	-0.37869542	0.19691266	-0.24546813	-0.14932929	0.2440648	-0.696762291
coryz	-0.31307985	-0.03978469	-0.67872906	0.47560095	-0.2717006	0.010100839
corxz	-0.14789379	0.35124150	0.60586509	0.49714739	-0.1725974	-0.293576958
medianx	0.45527279	0.07263250	-0.15107667	0.04118316	0.2041119	-0.271555352
specenergy	0.02626305	0.50763800	-0.20496822	-0.10762455	-0.1635708	0.287306862
varx	-0.05118606	0.48871957	-0.06373790	0.14172184	0.5956551	0.414531947
vary	0.34529931	0.29002244	-0.10431401	-0.39415833	-0.4780754	-0.118105967
varz	-0.18756231	0.47731475	0.02344709	-0.14063736	-0.2413950	-0.024514656
xbtwnzandy	0.42347512	0.04579115	-0.08478902	0.53628252	-0.2013935	-0.005568838
meanxsq	0.43860315	0.14287807	-0.15646326	0.09080863	0.2936192	-0.292450980

Figure 4 Loadings of Principal Components

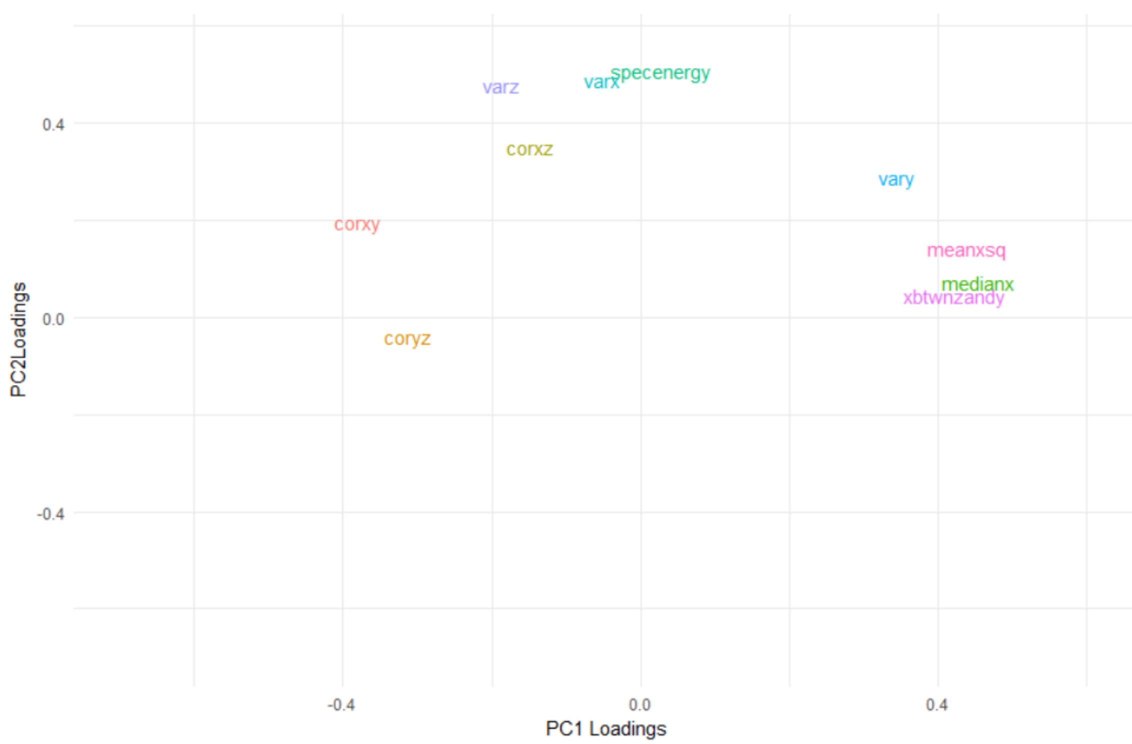

Figure 5 Loadings Plot

Performing a Naïve Bayes on the pca data gives a mean accuracy of 95.30%. which is almost a 5% increase from the Naïve Bayes model.

Predictions	Actual Climb_stairs	Drink_glass	Getup_bed	Row Total
Climb_stairs	20 0.267	0 0.000	1 0.013	21
Drink_glass	0 0.000	27 0.360	1 0.013	28
Getup_bed	1 0.013	0 0.000	25 0.333	26
Column Total	21	27	27	75

Figure 6 Naive Bayes on PCA

Visualizing the PCA components. We can see from the image below the with the first two principal components the three different activities are divided into fairly separate clusters with some overlap of the climb_stairs and getup_bed activities.

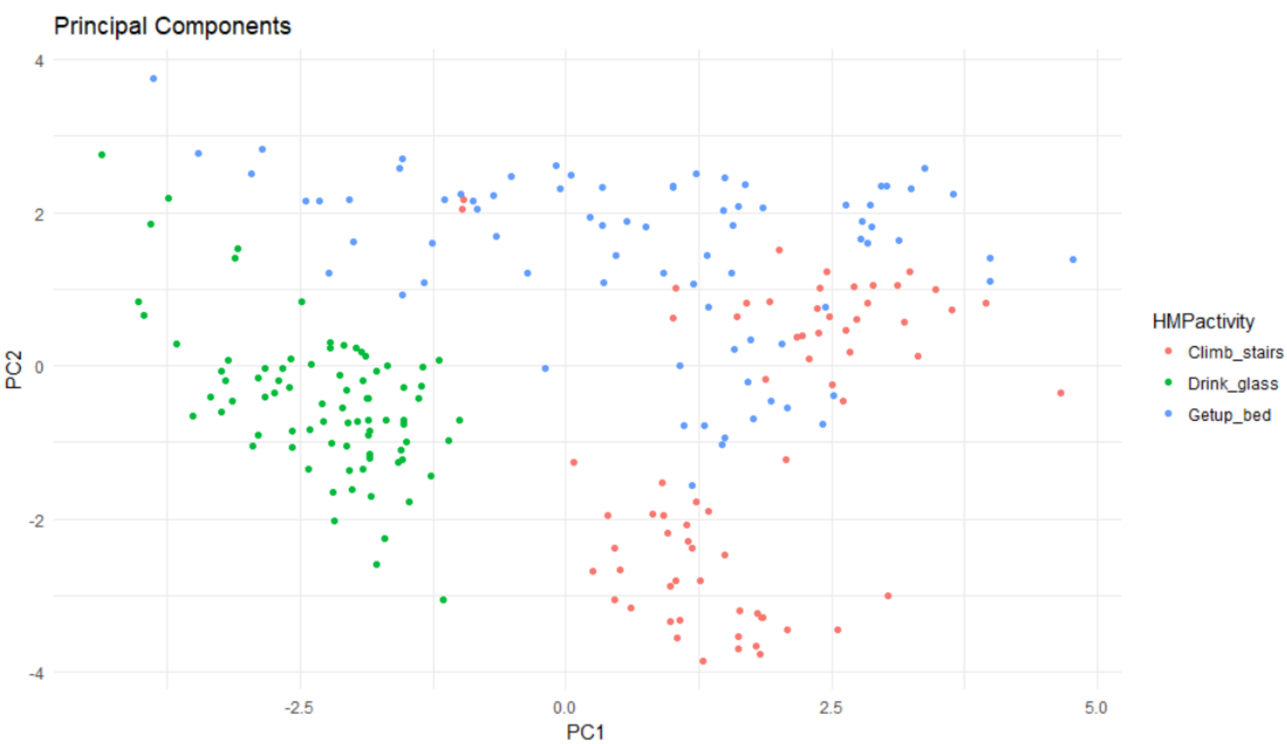

Figure 7 PCA plot

R Code:

```
setwd("D:/Files/Spring 18/Crypto/IOT/ADL_Dataset (1)/HMP_Dataset")

library(ggplot2)
library(gmodels)

dirlist <- list.dirs(recursive = F)[c(2,5,8)]
filelist <- sapply(dirlist,function(x) list.files(x))

resdf <- data.frame()
for (i in 1:length(dirlist)){

  for (j in 1:length(filelist[[i]])) {

  file <- paste0(paste0(dirlist[i],"/",filelist[[i]][j]))
  dat <- read.table(file)
  names(dat) <- c("x_acc","y_acc","z_acc")

  dattrimend <- dat[c(1:(nrow(dat)-80)),]
  cors <- cor(dat)

  variances <- var(dat)
  fftx <- fft(dat$x_acc)
  ffty <- fft(dat$y_acc)
  fftz <- fft(dat$z_acc)

  resdf <- rbind(resdf,data.frame(vol=gsub(".txt","",strsplit(filelist[[i]][j],"-")[[1]][9]),
                                  HMPactivity=gsub("./","",dirlist[i]),
                                  corxy=cors[1,2],
                                  coryz=cors[2,3],
                                  corxz=cors[1,3],
                                  medianx = median(dat$x_acc),
                                  #fftxmax=max(abs(Mod(fftx))), #Peak Amplitude of the Fourier
                                     Transform
                                  #fftymax=max(abs(Mod(ffty))),
                                  #fftzmax=max(abs(Mod(fftz))),

specenergy=(sum(Mod(fftx))/length(fftx))+(sum(Mod(ffty))/length(ffty))+(sum(Mod(fftz)
)/length(fftz)), #Spectral Energy
                                  varx=variances[1,1],
                                  vary=variances[2,2],
                                  varz=variances[3,3],
                                  #varxy=variances[1,2],
                                  #varyz=variances[2,3],
                                  #varxz=variances[1,3],
                                  xbtwnzandy=mean((dat$z_acc > dat$x & dat$y_acc < dat$x_acc)),
#drinkglasscharacteristic
```

```
                     #trimxbtwnzandy=mean((dattrimend$z_acc > dattrimend$x &
dattrimend$y_acc < dattrimend$x_acc)),
                     meanxsq = mean(dat$x_acc)**2
                     ))

  }
}

#Correlation Matrix
corrplot(cor(resdf[,-c(1,2)]),method = "square",type="lower")

library(e1071)

accs <- vector()

for (i in 1:1000) {

#Shuffle and Split dataset
resdf <- resdf[sample.int(nrow(resdf),nrow(resdf)),]
test <- resdf[sample(nrow(resdf),floor((25/100) * nrow(resdf))),]
train <- resdf[-sample(nrow(resdf),floor((25/100) * nrow(resdf))),]

#NaiveBayes
model <- naiveBayes(HMPactivity ~.,data = train)
preds <- predict(model,newdata = test)
tab <- table(preds,test$HMPactivity)
accs[i] <- round((sum(diag(tab))/sum(tab))*100,2)

}


#print(tab)
paste0("Accuracy is: ",mean(accs)," %")


##pca
library(caret)

pcamod <- preProcess(resdf[,-c(1,2)],method=c("BoxCox","center","scale","pca"),thresh
= .95)
loads <- pcamod$rotation #Loadings
pcadata <- predict(pcamod,resdf)
#Visualising the PCA

ggplot(pcadata)+geom_point(aes(PC1,PC2,color=HMPactivity),size=1.5)+theme_minimal()
+
  ggtitle("Principal Components")
```

```
#Visulaising Loadings
print(loads)


ggplot(data.frame(loads),aes(loads[,1],loads[,2],color=as.factor(c(1:length(dimnames(loads
)[[1]])))))+
    geom_text(label=dimnames(loads)[[1]],check_overlap = F,size=4)+
  theme_minimal()+
  theme(legend.position = "none")+xlab("PC1 Loadings")+ylab("PC2Loadings")+
xlim(c(min(loads),max(loads)))+
  ylim(c(min(loads),max(loads)))



##NaiveBayes on PCA Data.


accs <- vector()

for (i in 1:1000) {

  #Shuffle and Split dataset
  pcadata <- pcadata[sample.int(nrow(pcadata),nrow(pcadata)),]
  test <- pcadata[sample(nrow(pcadata),floor((25/100) * nrow(pcadata))),]
  train <- pcadata[-sample(nrow(pcadata),floor((25/100) * nrow(pcadata))),]

  #NaiveBayes
  model <- naiveBayes(HMPactivity ~.,data = train)
  preds <- predict(model,newdata = test)
  tab <- table(preds,test$HMPactivity)
  accs[i] <- round((sum(diag(tab))/sum(tab))*100,2)

}


#print(tab)
pasteo("Accuracy is: ",mean(accs)," %")
```