휴대폰 센서 데이터 추가 변수 추출

1.통계 특징과 관련된 변수 추출

- 기존에는 **roationRate**와 **userAcceleraion** 2개의 변수만 사용해서 통계 특징을 추출하였습니다.

(1) "**gravity**" 와 "**attitude**" 관련 변수들에 대한 통계 특징도 추출하였습니다.
***** ("gravity" 변수는 mag 함수를 적용하여 통계 특징을 추출하였고, "attitude" 변수는 기존 값에서 통계 특징을 추출하였습니다.)

- 기존에는 mag함수를 사용해서 x, y, z 크기를 구한 후 통계 특징을 추출하였습니다. 각각의 값을 그대로 사용해보는 것도 의미가 있을것이라는 생각이 들었습니다.

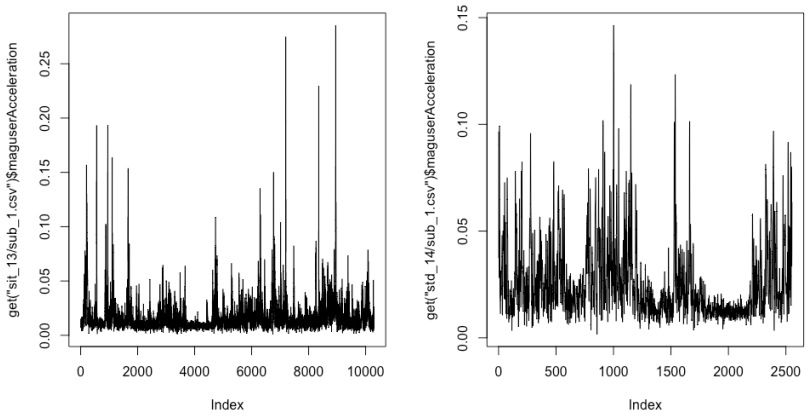
(2) Mag함수를 적용하지 않은 x, y, z 원래의 값에서 통계 특징을 추출하였습니다.
(3) (mean, min, max, sd, skewness, rms, rss, IQR, kurtosis) + **range** 통계 특징을 추가 하였습니다.
***** (range는 최대값과 최솟값의 차이로 dispersion 관련 통계 특징입니다.)

변수 명	변수 설명
변수명_range	각 변수에서 range 통계 특징 추출
attitude.roll_fn1 ~ attitude.roll_fn9	attitude.roll 변수에 (mean, min, max, sd, skewness, rms, rss, IQR, kurtosis) 통계 특징 추출
attitude.pitch_fn1 ~ attitude.pitch_fn9	attitude.pitch 변수에 (mean, min, max, sd, skewness, rms, rss, IQR, kurtosis) 통계 특징 추출
attitude.yaw_fn1 ~ attitude.yaw_fn9	attitude.yaw 변수에 (mean, min, max, sd, skewness, rms, rss, IQR, kurtosis) 통계 특징 추출
maggravity_fn1 ~ maggravity_fn9	Gravity 변수에 mag 함수를 취하여 (mean, min, max ,sd ,skewness, rms, rss, IQR, kurtosis) 통계 특징 추출
변수.x_fn1 ~ fn9, 변수.y_fn1~fn9, 변수.z_fn1~fn9	mag 함수를 적용하지 않은 상태에서 통계 특징 추출

2. 피크와 관련된 변수 추출

(1) 기존에는 피크 크기에서 (min, max, min, std) 통계 특징을 사용하였습니다. 여기에 **range** 통계 특징을 추가하였습니다.

(2) 기존에는 magrotationRate 변수의 피크만 가지고 변수를 도출하였습니다. 이 부분을 똑같이 **maguserAcceleration** 에도 적용을 하였습니다.



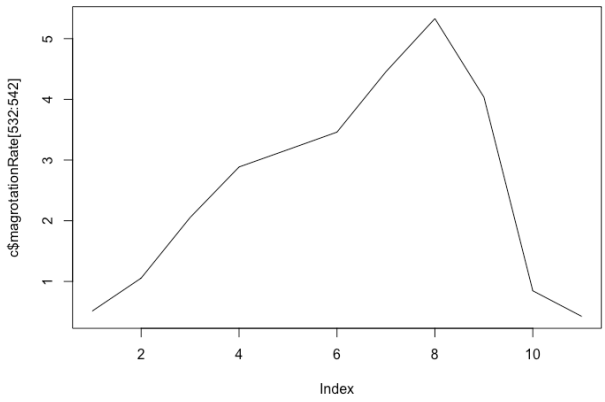
maguserAcceleration의 그래프를 그려본 후 **피크 기준값을 1**로 사용하기로 결정 하였습니다.

```
ex<- get(fls[1])
p_ro<- findpeaks(ex$magrotationRate, threshold=4)
```

```
> p_ro
          [,1] [,2] [,3] [,4]
 [1,] 7.138682  226  223  228
 [2,] 5.199968  304  298  307
 [3,] 4.996099  373  366  379
 [4,] 5.330121  539  532  542
 [5,] 5.599312  603  599  608
 [6,] 5.252531  614  608  617
 [7,] 5.303565  640  634  645
 [8,] 5.030706  850  845  855
 [9,] 5.696929  887  877  892
[10,] 5.007321  899  892  902
[11,] 5.020270  924  915  929
[12,] 4.588252  935  929  942
[13,] 5.124730 1232 1225 1235
[14,] 5.761030 1266 1259 1268
[15,] 5.152792 1288 1282 1293
[16,] 5.835136 1299 1293 1301
[17,] 5.664433 1576 1571 1579
[18,] 6.239476 1627 1623 1633
```

다음과 같이 예제로 피크를 추출하고, 피크, 피크 발생 전, 피크 발생 후에 관한 그래프를 그려보았습니다.

```
plot(ex$magrotationRate[532:542], type="l")
```



(3) 피크가 **발생하기 전과 후**의 값도 의미가 있을 것이라고 생각하여 피크가 발생하기 바로 이전 값과 이후 값을 구하여 (min, max, min, std) 통계 특징을 구한 변수를 추가하였습니다.

(4) **순간적인 변화**를 알기 위해 피크가 발생하는 순간과 그 다음 순간의 차이를 구하여 (min, max, min, std) 통계 특징을 구한 변수를 추가하였습니다.

변수 명	변수 설명
p_ac_n, p_ac_interval, p_ac_interval_std p_ac_mean, p_ac_max, p_ac_min, p_ac_std	magrotationRate로 도출했던 피크 특징들을 maguserAcceleration에 똑같이 적용
p_ro_be_mean, p_ro_be_max, p_ro_be_min, p_ro_be_std, p_ro_af_mean, p_ro_af_max, p_ro_ad_min, p_ro_af_std p_ac_be_mean, p_ac_be_max, p_ac_be_min, p_ac_be_std, p_ac_af_mean, p_ac_af_max, p_ac_af_min, p_ac_af_std	magrotationRate과 maguserAcceleration 변수에서 피크값을 구한 후 피크 바로 이전 값과 이후 값을 구해 (min, max, min, std) 통계 특징을 추출
p_ro_range, p_ac_range	피크 크기를 구하여 range 통계 특징 추출
p_ro_moment_mean, p_ro_moment_max, p_ro_moment_min, p_ro_moment_std, p_ac_moment_mean, p_ac_moment_max, p_ac_moment_min, p_ac_moment_std,	magrotationRate과 maguserAcceleration 변수에서 피크값과 바로 이후 값의 차이를 구한 후 (min, max, min, std) 통계 특징을 추출

3.고속 푸리 변환 적용 후 통계 특징 변수 추출

fft 함수로 푸리에 변환을 적용해 데이터를 변환하였습니다. 변환 후 실수 부분만 가져와서 통계 특징을 추출하였습니다.

퓨리 변환 적용

attitude.roll	attitude.pitch
-7550.155337+0i	-5345.562433+0i
78.929731652836-818.21028714885i	-140.415666199963+75.008275724803i
-3.192766641088+400.151604055224i	-4.8393848124534+20.5110945156602i
57.496678634217-199.296855887161i	-13.67909670239+11.2172647845625i
15.202973547951+189.920905751619i	-16.9188658378404+28.4679725126 75i
118.741768976468-151.1699058889 59i	-6.90234803530123-8.44747908113383i

실수 부분 추출

attitude.roll	attitude.pitch
-7550.155337	-5.345562e+03
78.929732	-1.404157e+02
-3.192767	-4.839385e+00
57.496679	-1.367910e+01
15.202974	-1.691887e+01
118.741769	-6.902348e+00

변수 명	변수 설명
변수명_fft1 ~ 변수명_fft9	먼저, 퓨리의 변한 함수인 fft()를 사용하여 데이터를 변환시킨 후 실수 부분만 선택하여 데이터를 만듦 그리고 각 변수에 (mean, min, max, sd, skewness, rms, rss, IQR, kurtosis, **range**) 통계 특징을 추출함

4.변화 분석과 관련된 변수 추출

- 기존에는 cpt.mean, cpt.var, cpt.meanvar에서 변화 분석을 진행할 때 method 파라미터를 기본값인 AMOC(single changepoint)를 사용하였습니다.

(1) method 방식 중 **PELT(multiple changepoints)** 방법이 좀 더 특징을 잘 추출하는 것으로 나타나 method를 변경하여 변화 분석을 진행하였습니다.

```
> e_extend
=== 10 Fold Cross Validation ===

=== Summary ===

Correctly Classified Instances         146               40.5556 %
Incorrectly Classified Instances       214               59.4444 %
Kappa statistic                          0.2793
K&B Relative Info Score                 37.4127 %
K&B Information Score                  344.3493 bits      0.9565 bits/instance
Class complexity | order 0             920.4079 bits      2.5567 bits/instance
Class complexity | scheme            12411.5555 bits     34.4765 bits/instance
Complexity improvement     (Sf)     -11491.1476 bits    -31.9199 bits/instance
Mean absolute error                      0.2155
Root mean squared error                  0.3381
Relative absolute error                 78.17   %
Root relative squared error             91.0718 %
Total Number of Instances              360
```

```
> e_extend
=== 10 Fold Cross Validation ===

=== Summary ===

Correctly Classified Instances         213               59.1667 %
Incorrectly Classified Instances       147               40.8333 %
Kappa statistic                          0.5059
K&B Relative Info Score                 56.8757 %
K&B Information Score                  523.4881 bits      1.4541 bits/instance
Class complexity | order 0             920.4079 bits      2.5567 bits/instance
Class complexity | scheme              424.7638 bits      1.1799 bits/instance
Complexity improvement     (Sf)        495.6441 bits      1.3768 bits/instance
Mean absolute error                      0.1581
Root mean squared error                  0.2901
Relative absolute error                 57.3452 %
Root relative squared error             78.1369 %
```

RWeka에 RF로 실험을 진행해 봤을때 AMOC(왼쪽)를 사용했을때 보다 PELT(오른쪽)를 사용했을때 더 높은 정확도를 가졌습니다.

- 기존에는 변화가 몇 번 일어났는지에 관한 변수만 사용하였습니다.

(2) 변화가 일어났던 위치의 값을 가져와서 (mean, max, min, std) 통계 특징을 추출한 변수를 추가하였습니다.

변수 명	변수 설명
cp1_mean, cp1_max, cp1_min, cp1_std, cp2_mean, cp2_max, cp2_min, cp2_std, cp3_mean, cp3_max, cp3_min, cp3_std, cp4_mean, cp4_max, cp4_min, cp4_std, cp5_mean, cp5_max, cp5_min, cp5_std, cp6_mean, cp6_max, cp6_min, cp6_std	*method 방식을 PELT를 사용하여 분석을 진행하였습니다. 변화가 일어났던 위치의 값을 가져와서 (mean, max, min, std) 통계 특징을 추출함

5.모델 결과

5-1 변수 추출 특징 별 모델 비교

xgboost 모델을 사용해서 학습을 진행하였습니다.

[통계 특징 변수 모델]

```
> Static_conf
Confusion Matrix and Statistics

          Reference
Prediction dws jog sit std ups wlk
       dws  69   1   0   1   2   2
       jog   0  47   0   0   0   0
       sit   1   0  48   0   0   0
       std   0   0   0  46   0   0
       ups   2   0   0   0  70   0
       wlk   0   0   0   1   0  70

Overall Statistics

               Accuracy : 0.9722
                 95% CI : (0.9495, 0.9866)
    No Information Rate : 0.2
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9664

 Mcnemar's Test P-Value : NA
```

- 통계 특징 변수만 사용했을 때 정확도는 **약 97%**가 나왔습니다. 아주 높은 정확도가 나왔습니다. 다른 변수들에 비해 중요한 변수들이 많이 담겨있는 것 같습니다.

[피크 특징 변수 모델]

```
> Peak_conf
Confusion Matrix and Statistics

          Reference
Prediction dws jog sit std ups wlk
       dws  55   1   0   0  13   7
       jog   2  44   1   0   1   1
       sit   0   0  36  13   0   0
       std   0   0  11  35   0   0
       ups  12   1   0   0  51   6
       wlk   3   2   0   0   7  58

Overall Statistics

               Accuracy : 0.775
                 95% CI : (0.7283, 0.8171)
    No Information Rate : 0.2
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7278

 Mcnemar's Test P-Value : NA
```

피크 특징 변수만 사용했을 때 정확도는 **약 78%**가 나왔습니다. 통계 특징보다는 낮은 결과지만 그래도 꽤 높은 정확도가 나왔다고 생각합니다.

[푸리에 변환 변수 모델]

```
> Fourier_conf
Confusion Matrix and Statistics

          Reference
Prediction dws jog sit std ups wlk
       dws  68   2   0   0   4   1
       jog   0  42   0   0   0   2
       sit   0   0  47   0   0   1
       std   0   0   1  48   0   0
       ups   3   0   0   0  68   0
       wlk   1   4   0   0   0  68

Overall Statistics

               Accuracy : 0.9472
                 95% CI : (0.9188, 0.9679)
    No Information Rate : 0.2
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9361

 Mcnemar's Test P-Value : NA
```

푸리에 변환을 적용한 변수를 사용했을 때 정확도는 **약 95%**가 나왔습니다. 높은 정확도가 나왔습니다.

[변화 분석 변수 모델]

```
> Chpoint_conf
Confusion Matrix and Statistics

          Reference
Prediction dws jog sit std ups wlk
       dws  52   1   0   0  14  11
       jog   1  37   0   0   0   4
       sit   0   0  45   4   0   0
       std   0   0   3  44   0   0
       ups  14   0   0   0  51   4
       wlk   5  10   0   0   7  53

Overall Statistics

               Accuracy : 0.7833
                 95% CI : (0.7371, 0.8248)
    No Information Rate : 0.2
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7376

 Mcnemar's Test P-Value : NA
```

변화 분석 변수를 사용했을 때 정확도는 **약 78%**가 나왔습니다. 수업시간에 실습을 하며 40%밖에 나오지 못해 좋은 예측 변수가 될 수 없다고 생각하였지만, method방식을 바꾸고, 통계 특징을 추출하니 꽤 높은 정확도가 나왔습니다.

5-2 최종 모델 선택 기준

(1) 모든 변수들을 통합한 모델
(2) 가장 정확도가 높았던 두 특징(통계 특징 변수, 푸리에 변환 변수) 변수를 사용한 모델
(3) 각 변수의 특징에서 중요 변수를 상위 30개씩 추출하여 합한 모델

저는 다음과 같이 3개의 모델로 모델링을 진행하여 정확도가 가장 높은 모델을 최종으로 선택하고 나머지 2개의 모델을 후보로 선택하였습니다.

5-3 최종 모델 선택

[모든 변수들을 통합한 모델]

```
> all_conf
Confusion Matrix and Statistics

          Reference
Prediction dws jog sit std ups wlk
       dws  69   1   0   0   2   1
       jog   0  47   0   0   0   0
       sit   0   0  48   0   0   0
       std   0   0   0  48   0   0
       ups   3   0   0   0  70   0
       wlk   0   0   0   0   0  71

Overall Statistics

               Accuracy : 0.9806
                 95% CI : (0.9603, 0.9921)
    No Information Rate : 0.2
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9765

 Mcnemar's Test P-Value : NA
```

- 모든 변수들을 통합한 모델은 정확도가 약 **98%**가 나왔습니다. 앞에서 가장 높았던 통계 특징만 추출한 모델보다 약 0.01% 증가하였습니다.

[가장 정확도가 높았던 두 특징(통계 특징 변수, 푸리에 변환 변수) 변수를 사용한 모델]

```
> Sta_Fo_conf
Confusion Matrix and Statistics

          Reference
Prediction dws jog sit std ups wlk
       dws  69   1   0   0   2   1
       jog   0  47   0   0   0   0
       sit   0   0  48   0   0   0
       std   0   0   0  48   0   0
       ups   2   0   0   0  70   0
       wlk   1   0   0   0   0  71

Overall Statistics

               Accuracy : 0.9806
                 95% CI : (0.9603, 0.9921)
    No Information Rate : 0.2
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9765

 Mcnemar's Test P-Value : NA
```

- 위에 모델과 동일한 결과가 나왔습니다. 이것으로 보아 역시 통계 특징에 관한 변수가 중요한 변수들인 것 같습니다.

[각 변수의 특징에서 중요 변수를 상위 30개씩 추출하여 합한 모델]

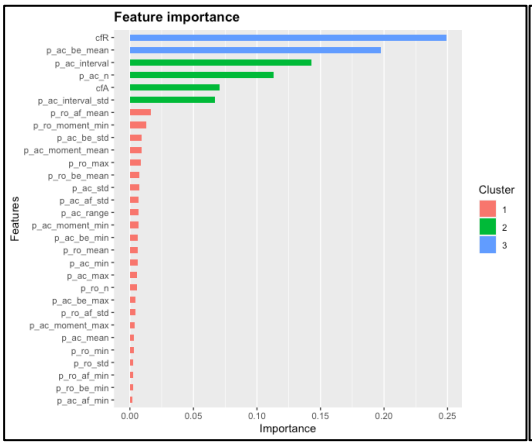
변수별 중요도를 파악하기 위해서는 xgboost로 학습을 시킨 후, xgb.importance 사용해서 확인할 수 있습니다.

```
> import_conf
Confusion Matrix and Statistics

          Reference
Prediction dws jog sit std ups wlk
       dws  68   1   0   0   2   1
       jog   0  47   0   0   0   0
       sit   0   0  48   0   0   0
       std   0   0   0  48   0   0
       ups   3   0   0   0  70   0
       wlk   1   0   0   0   0  71

Overall Statistics

               Accuracy : 0.9778
                 95% CI : (0.9567, 0.9904)
    No Information Rate : 0.2
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9731

 Mcnemar's Test P-Value : NA
```

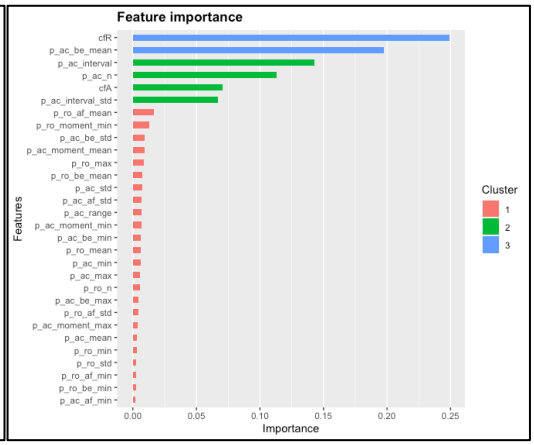
- 정확도는 약 98%가 나왔습니다. 위의 두 모델보다는 아주 약간 떨어지는 결과가 나왔습니다.

각 변수별 중요도는 xgb.ggplot.importance 함수를 사용해서 다음과 같이 시각화도 해볼 수 있습니다.

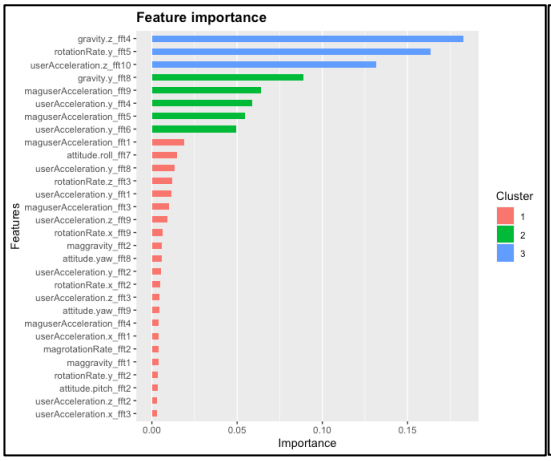
[통계 특징 중요 변수]



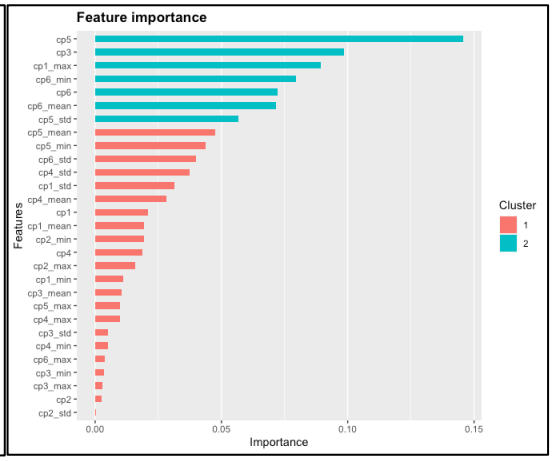
[피크 특징 중요 변수]



[푸리에 변환 특징 중요 변수]



[변화 분석 특징 중요 변수]



위 3가지 모델의 결과를 비교해본 결과 최종 모델로 **"모든 변수를 통합한 모델"** 을 선택하였습니다. 통계적 모델과 푸리에 변환 모델을 합한 모델도 정확도가 높았지만, 여러 특징을 담은 모델을 사용하고 싶어 이 모델을 선택하였습니다.

[코드 부록]

```
## library load
library(dplyr)
library(stringr)
library(ggplot2)
library(tidyverse)
library(RWeka)
library(fBasics)
library(pracma)
library(signal)
library(seewave)
library(e1071)
library(caret)
library(xgboost)
library(changepoint)

# 경로 설정
setwd("/Users/seominji/Desktop/Unstruct_DA/A_DeviceMotion_data")
d<-getwd()
fls <-dir(d,recursive = TRUE)

# 객체 생성
for(f in fls){
  a<- file.path(str_c(d,"/",f))
  temp<- read.csv(a)
  assign(f,temp)
}

#### 통계치 관련 변수 ####

# 전체 데이터 생성(피크변수와 merge를 안전하게 하기 위해 d(파일 이름)을 추가)
HAR_total<-data.frame()
i<-0
for(f in fls){
  temp<-get(f)
  print(f)
  i<-i+1
  print(i)

  HAR_total <-rbind(HAR_total,
                    temp%>%mutate(exp_no=unlist(regmatches(f,gregexpr("[[:digit:]]+", f)[1]))[1],
                                  id=unlist(regmatches(f,gregexpr("[[:digit:]]+", f)[1]))[2],
                                  activity=unlist(str_split(f,"₩₩_"))[1],d=f))
}

# mag 함수 정의
mag<- function(df, column){
```

```
  df[,str_c("mag", column)]<- with(df, sqrt(get(str_c(column, ".x"))^2+get(str_c(column,".y"))^2+get(str_c(column, ".z"))^2))
  return(df)
}

# skewness 함수
skewness <- function(x){
  (sum((x-mean(x))^3)/length(x))/((sum((x-mean(x))^2)/length(x)))^(3/2)
}

# rss 함수 정의
rss<-function(x) rms(x)*(length(x))*0.5

# range 함수 정의
range_ <- function(x){
  (diff(range(x)))
}

# mag 적용(gravity 추가)
HAR_total<- mag(HAR_total, "userAcceleration")
HAR_total<- mag(HAR_total, "rotationRate")
HAR_total<-mag(HAR_total, "gravity")

## 변수 추출 ##
# 통계 특징 구하기("maggravity","attitude.roll","attitude.pitch","attitude.yaw" 추가)
HAR_summary_extend<- HAR_total %>% group_by(id,exp_no,activity,d) %>%
  summarise_at(.vars=c("gravity.x","gravity.y","gravity.z",
                       "userAcceleration.x","userAcceleration.y","userAcceleration.z",
                       "rotationRate.x","rotationRate.y","rotationRate.z",
                       "maguserAcceleration","magrotationRate",
                       "maggravity","attitude.roll","attitude.pitch","attitude.yaw"),
               .funs=c(mean, min, max ,sd ,skewness, rms, rss, IQR, e1071::kurtosis, range_))

# 널값 확인
colSums(is.na(HAR_summary_extend))

## 통계 특징 변수만 학습 ##
# x,y 구분
x = HAR_summary_extend %>% ungroup %>% select(-d, -exp_no, -id, -activity) %>% data.matrix
y = HAR_summary_extend$activity

# xgboost 모델 학습(10-fold)
set.seed(1004)
Static_model = xgb.cv(data = x,label = as.integer(as.factor(y))-1, num_class = levels(as.factor(y)) %>% length,
                      nfold = 10, nrounds = 500, early_stopping_rounds = 8, booster = 'gbtree',
                      objective = 'multi:softprob', eval_metric = 'mlogloss',
                      verbose = F, prediction = T)

# 예측 데이터 생성(max.col 사용해서 가장 큰값 추출)
pred_df = Static_model$pred %>% as.data.frame %>%
```

```
  mutate(pred = levels(as.factor(y))[max.col(.)] %>% as.factor, actual = as.factor(y))

pred_df %>% select(pred,actual) %>% table

# confusionMatrix 생성
Static_conf<- caret::confusionMatrix(pred_df$pred, pred_df$actual)
Static_conf

# 중요 변수 추출
Static_model = xgboost(data = x,label = as.integer(as.factor(y))-1, num_class = levels(as.factor(y)) %>% length,
                       nfold = 10, nrounds = 500, early_stopping_rounds = 8, booster = 'gbtree',
                       objective = 'multi:softprob', eval_metric = 'mlogloss',
                       verbose = F)

Static_imp<- xgb.importance(model = Static_model)


#### ####

#### 피크 관련 변수 ####
Peak_rslt<-data.frame()

for(d in fls){
  f<-get(d)
  f<-mag(f,"rotationRate")
  f<-mag(f,"userAcceleration")
  assign(d,f)}# d는 객체명 f는 객체가 들어간다

## 변수 추출 ##
for(d in fls){
  f<-get(d)

  p_ro<-findpeaks(f$magrotationRate,threshold = 4)
  p_ac<-findpeaks(f$maguserAcceleration,threshold = 1)
  Peak_rslt<-rbind(Peak_rslt, data.frame(d,
  p_ro_n=ifelse(!is.null(p_ro),dim(p_ro)[1],0),
  p_ro_interval=ifelse(!is.null(p_ro),ifelse(dim(p_ro)[1]>2,mean(diff(p_ro[,2])),0),0),
  p_ro_interval_std=ifelse(!is.null(p_ro),ifelse(dim(p_ro)[1]>2,std(diff(p_ro[,2])),0),0),
  p_ro_mean=ifelse(!is.null(p_ro),mean(p_ro[,1]),0),
  p_ro_max=ifelse(!is.null(p_ro),max(p_ro[,1]),0),
  p_ro_min=ifelse(!is.null(p_ro),min(p_ro[,1]),0),
  p_ro_range=ifelse(!is.null(p_ro),diff(range(p_ro[,1])),0),
  p_ro_std=ifelse(!is.null(p_ro),std(p_ro[,1]),0),
  p_ro_be_mean=ifelse(!is.null(p_ro),mean(f$magrotationRate[p_ro[,2]-1]),0),
  p_ro_be_max=ifelse(!is.null(p_ro),max(f$magrotationRate[p_ro[,2]-1]),0),
  p_ro_be_min=ifelse(!is.null(p_ro),min(f$magrotationRate[p_ro[,2]-1]),0),
  p_ro_be_std=ifelse(!is.null(p_ro),std(f$magrotationRate[p_ro[,2]-1]),0),
  p_ro_af_mean=ifelse(!is.null(p_ro),mean(f$magrotationRate[p_ro[,2]+1]),0),
  p_ro_af_max=ifelse(!is.null(p_ro),max(f$magrotationRate[p_ro[,2]+1]),0),
```

```
  p_ro_af_min=ifelse(!is.null(p_ro),min(f$magrotationRate[p_ro[,2]+1]),0),
  p_ro_af_std=ifelse(!is.null(p_ro),std(f$magrotationRate[p_ro[,2]+1]),0),
  p_ro_moment_mean=ifelse(!is.null(p_ro),mean(p_ro[,1] - f$magrotationRate[p_ro[,2]+1]),0),
  p_ro_moment_max=ifelse(!is.null(p_ro),max(p_ro[,1] - f$magrotationRate[p_ro[,2]+1]),0),
  p_ro_moment_min=ifelse(!is.null(p_ro),min(p_ro[,1] - f$magrotationRate[p_ro[,2]+1]),0),
  p_ro_moment_std=ifelse(!is.null(p_ro),std(p_ro[,1] - f$magrotationRate[p_ro[,2]+1]),0),
  p_ac_n=ifelse(!is.null(p_ac),dim(p_ac)[1],0),
  p_ac_interval=ifelse(!is.null(p_ac),ifelse(dim(p_ac)[1]>2,mean(diff(p_ac[,2])),0),0),
  p_ac_interval_std=ifelse(!is.null(p_ac),ifelse(dim(p_ac)[1]>2,std(diff(p_ac[,2])),0),0),
  p_ac_mean=ifelse(!is.null(p_ac),mean(p_ac[,1]),0),
  p_ac_max=ifelse(!is.null(p_ac),max(p_ac[,1]),0),
  p_ac_min=ifelse(!is.null(p_ac),min(p_ac[,1]),0),
  p_ac_range=ifelse(!is.null(p_ac),diff(range(p_ac[,1])),0),
  p_ac_std=ifelse(!is.null(p_ac),std(p_ac[,1]),0),
  p_ac_be_mean=ifelse(!is.null(p_ac),mean(f$maguserAcceleration[p_ac[,2]-1]),0),
  p_ac_be_max=ifelse(!is.null(p_ac),max(f$maguserAcceleration[p_ac[,2]-1]),0),
  p_ac_be_min=ifelse(!is.null(p_ac),min(f$maguserAcceleration[p_ac[,2]-1]),0),
  p_ac_be_std=ifelse(!is.null(p_ac),std(f$maguserAcceleration[p_ac[,2]-1]),0),
  p_ac_af_mean=ifelse(!is.null(p_ac),mean(f$maguserAcceleration[p_ac[,2]+1]),0),
  p_ac_af_max=ifelse(!is.null(p_ac),max(f$maguserAcceleration[p_ac[,2]+1]),0),
  p_ac_af_min=ifelse(!is.null(p_ac),min(f$maguserAcceleration[p_ac[,2]+1]),0),
  p_ac_af_std=ifelse(!is.null(p_ac),std(f$maguserAcceleration[p_ac[,2]+1]),0),
  p_ac_moment_mean=ifelse(!is.null(p_ac),mean(p_ac[,1] - f$maguserAcceleration[p_ac[,2]+1]),0),
  p_ac_moment_max=ifelse(!is.null(p_ac),max(p_ac[,1] - f$maguserAcceleration[p_ac[,2]+1]),0),
  p_ac_moment_min=ifelse(!is.null(p_ac),min(p_ac[,1] - f$maguserAcceleration[p_ac[,2]+1]),0),
  p_ac_moment_std=ifelse(!is.null(p_ac),std(p_ac[,1] - f$maguserAcceleration[p_ac[,2]+1]),0)

  ))}

# 파고율 변수
temp<- data.frame()
for(d in fls){
  f<-get(d)
  f<-f %>% select(magrotationRate, maguserAcceleration)
  cfR<- crest(f$magrotationRate, 50, plot=TRUE)
  cfA<- crest(f$maguserAcceleration, 50, plot=TRUE)
  temp<- rbind(temp, data.frame(d, cfR=cfR$C, cfA=cfA$C))
}

Peak_final<- merge(Peak_rslt, temp, by="d")

# exp, id, activity 추출
id_f<-function(x){
  exp_no=unlist(regmatches(x,gregexpr("[[:digit:]]+", x)[1]))[1]
  id=unlist(regmatches(x,gregexpr("[[:digit:]]+", x)[1]))[2]
  activity=unlist(str_split(x, "₩₩_"))[1]
  return(cbind(exp_no, id, activity))
}
```

```
temp<-data.frame()
for(i in 1:nrow(Peak_final)){
  temp<-rbind(temp, id_f(Peak_final$d[i]))
}

Peak_final2<-cbind(Peak_final,temp)

## 피크 특징 변수만 학습 ##
# x,y 구분
x = Peak_final2 %>% ungroup %>% select(-d, -exp_no, -id, -activity) %>% data.matrix
y = Peak_final2$activity

# xgboost 모델 학습(10-fold)
set.seed(1004)

Peak_model = xgb.cv(data = x,label = as.integer(as.factor(y))-1, num_class = levels(as.factor(y)) %>% length,
                    nfold = 10, nrounds = 500, early_stopping_rounds = 8, booster = 'gbtree',
                    objective = 'multi:softprob', eval_metric = 'mlogloss',
                    verbose = F, prediction = T)

# 예측 데이터 생성(max.col 사용해서 가장 큰값 추출)
pred_df = Peak_model$pred %>% as.data.frame %>%
  mutate(pred = levels(as.factor(y))[max.col(.)] %>% as.factor, actual = as.factor(y))

pred_df %>% select(pred,actual) %>% table

# confusionMatrix 생성
Peak_conf<- caret::confusionMatrix(pred_df$pred, pred_df$actual)
Peak_conf

# 중요 변수 추출
Peak_model = xgboost(data = x,label = as.integer(as.factor(y))-1, num_class = levels(as.factor(y)) %>% length,
                     nfold = 10, nrounds = 500, early_stopping_rounds = 8, booster = 'gbtree',
                     objective = 'multi:softprob', eval_metric = 'mlogloss',
                     verbose = F)

Peak_imp<- xgb.importance(model = Peak_model)


#### ####

####고속 푸리에 변환 (Fast Fourier transform) 적용 변수 ####
fft_data <- data.frame()
for(f in fls){
  temp<-get(f)

  temp<- as.data.frame(lapply(temp, fft)) # fft 적용
  temp<- as.data.frame(lapply(temp, Re)) # 앞에 실수부분만 가져옴
```

```
  fft_data <-rbind(fft_data,
                   temp%>%mutate(exp_no=unlist(regmatches(f,gregexpr("[[:digit:]]+", f)[1]))[1],
                                 id=unlist(regmatches(f,gregexpr("[[:digit:]]+", f)[1]))[2],
                                 activity=unlist(str_split(f,"₩₩_"))[1],d=f))
}

fft_data<- mag(fft_data, "userAcceleration")
fft_data<- mag(fft_data, "rotationRate")
fft_data<-mag(fft_data, "gravity")
colnames(fft_data)

## 변수 추출 ##
fft_data_summary<- fft_data %>% group_by(id,exp_no,activity,d) %>%
  summarise_at(.vars=c("gravity.x","gravity.y","gravity.z",
                       "userAcceleration.x","userAcceleration.y","userAcceleration.z",
                       "rotationRate.x","rotationRate.y","rotationRate.z",
                       "maguserAcceleration","magrotationRate",
                       "maggravity","attitude.roll","attitude.pitch","attitude.yaw"),
               .funs=c(mean, min, max ,sd ,skewness, rms, rss, IQR, e1071::kurtosis, range_))

# 변수명 변경
names(fft_data_summary)<- str_replace_all(colnames(fft_data_summary),'fn',"fft")

## FFT 변환 변수만 학습 ##
# x,y 구분
x = fft_data_summary %>% ungroup %>% select(-d, -exp_no, -id, -activity) %>% data.matrix
y = fft_data_summary$activity

# xgboost 모델 학습(10-fold)
set.seed(1004)
fft_model = xgb.cv(data = x,label = as.integer(as.factor(y))-1, num_class = levels(as.factor(y)) %>% length,
                   nfold = 10, nrounds = 500, early_stopping_rounds = 8, booster = 'gbtree',
                   objective = 'multi:softprob', eval_metric = 'mlogloss',
                   verbose = F, prediction = T)

# 예측 데이터 생성(max.col 사용해서 가장 큰값 추출)
pred_df = fft_model$pred %>% as.data.frame %>%
  mutate(pred = levels(as.factor(y))[max.col(.)] %>% as.factor, actual = as.factor(y))

pred_df %>% select(pred,actual) %>% table

# confusionMatrix 생성
Fourier_conf<- caret::confusionMatrix(pred_df$pred, pred_df$actual)
Fourier_conf

# 중요 변수 추출
Fourier_model = xgboost(data = x,label = as.integer(as.factor(y))-1, num_class = levels(as.factor(y)) %>% length,
                        nfold = 10, nrounds = 500, early_stopping_rounds = 8, booster = 'gbtree',
                        objective = 'multi:softprob', eval_metric = 'mlogloss',
```

```
                        verbose = F)

Fourier_imp<- xgb.importance(model = Fourier_model)

#### ####

#### 변화 분석 변수 ####
## 변수 추출
ch_pt<-data.frame()

for(d in fls){
  f<-get(d)
  f<-mag(f, "rotationRate")
  f<-mag(f, "userAcceleration")
  f<-mag(f, "gravity")# method 값을 PELT 방식으로 바꿈(기본값은 AMOC)
  rslt<-sapply(f %>% select(magrotationRate, maguserAcceleration, maggravity), cpt.mean, method = "PELT")
  rslt_cpts1<-cpts(rslt$magrotationRate)
  # 변화시점에 해당하는 값을 가져와 통계 특징을 추출
  cp1_mean<- ifelse(length(rslt_cpts1) != 0, mean(f$magrotationRate[rslt_cpts1]),0)
  cp1_max<- ifelse(length(rslt_cpts1) != 0, max(f$magrotationRate[rslt_cpts1]),0)
  cp1_min<- ifelse(length(rslt_cpts1) != 0, min(f$magrotationRate[rslt_cpts1]),0)
  cp1_std<- ifelse(length(rslt_cpts1) != 0, std(f$magrotationRate[rslt_cpts1]),0)
  rslt_cpts2<-cpts(rslt$maguserAcceleration)
  cp2_mean<- ifelse(length(rslt_cpts2) != 0, mean(f$maguserAcceleration[rslt_cpts2]),0)
  cp2_max<- ifelse(length(rslt_cpts2) != 0, max(f$maguserAcceleration[rslt_cpts2]),0)
  cp2_min<- ifelse(length(rslt_cpts2) != 0, min(f$maguserAcceleration[rslt_cpts2]),0)
  cp2_std<- ifelse(length(rslt_cpts2) != 0, std(f$maguserAcceleration[rslt_cpts2]),0)
  rslt2<-sapply(f %>% select(magrotationRate, maguserAcceleration), cpt.var, method = "PELT")
  rslt2_cpts1<-cpts(rslt2$magrotationRate)
  cp3_mean<- ifelse(length(rslt2_cpts1) != 0, mean(f$magrotationRate[rslt2_cpts1]),0)
  cp3_max<- ifelse(length(rslt2_cpts1) != 0, max(f$magrotationRate[rslt2_cpts1]),0)
  cp3_min<- ifelse(length(rslt2_cpts1) != 0, min(f$magrotationRate[rslt2_cpts1]),0)
  cp3_std<- ifelse(length(rslt2_cpts1) != 0, std(f$magrotationRate[rslt2_cpts1]),0)
  rslt2_cpts2<-cpts(rslt2$maguserAcceleration)
  cp4_mean<- ifelse(length(rslt2_cpts2) != 0, mean(f$maguserAcceleration[rslt2_cpts2]),0)
  cp4_max<- ifelse(length(rslt2_cpts2) != 0, max(f$maguserAcceleration[rslt2_cpts2]),0)
  cp4_min<- ifelse(length(rslt2_cpts2) != 0, min(f$maguserAcceleration[rslt2_cpts2]),0)
  cp4_std<- ifelse(length(rslt2_cpts2) != 0, std(f$maguserAcceleration[rslt2_cpts2]),0)
  rslt3<-sapply(f %>% select(magrotationRate, maguserAcceleration), cpt.meanvar, method = "PELT")
  rslt3_cpts1<-cpts(rslt3$magrotationRate)
  cp5_mean<- ifelse(length(rslt3_cpts1) != 0, mean(f$magrotationRate[rslt3_cpts1]),0)
  cp5_max<- ifelse(length(rslt3_cpts1) != 0, max(f$magrotationRate[rslt3_cpts1]),0)
  cp5_min<- ifelse(length(rslt3_cpts1) != 0, min(f$magrotationRate[rslt3_cpts1]),0)
  cp5_std<- ifelse(length(rslt3_cpts1) != 0, std(f$magrotationRate[rslt3_cpts1]),0)
  rslt3_cpts2<-cpts(rslt3$maguserAcceleration)
  cp6_mean<- ifelse(length(rslt3_cpts2) != 0, mean(f$maguserAcceleration[rslt3_cpts2]),0)
  cp6_max<- ifelse(length(rslt3_cpts2) != 0, max(f$maguserAcceleration[rslt3_cpts2]),0)
  cp6_min<- ifelse(length(rslt3_cpts2) != 0, min(f$maguserAcceleration[rslt3_cpts2]),0)
  cp6_std<- ifelse(length(rslt3_cpts2) != 0, std(f$maguserAcceleration[rslt3_cpts2]),0)
```

```
  ch_pt<-rbind(ch_pt, data.frame(d, cp1=length(rslt_cpts1),cp2=length(rslt_cpts2),
                                 cp3=length(rslt2_cpts1), cp4=length(rslt2_cpts2),
                                 cp5=length(rslt3_cpts1), cp6=length(rslt3_cpts2),
                                 cp1_mean, cp1_max, cp1_min, cp1_std,
                                 cp2_mean, cp2_max, cp2_min, cp2_std,
                                 cp3_mean, cp3_max, cp3_min, cp3_std,
                                 cp4_mean, cp4_max, cp4_min, cp4_std,
                                 cp5_mean, cp5_max, cp5_min, cp5_std,
                                 cp6_mean, cp6_max, cp6_min, cp6_std))
}


for(d in fls){
  f<-get(d)
  f<-mag(f,"rotationRate")
  f<-mag(f,"userAcceleration")
  assign(d,f)}# d는 객체명 f는 객체야!!!

# exp, id, activity 추출
id_f<-function(x){
  exp_no=unlist(regmatches(x,gregexpr("[[:digit:]]+", x)[1]))[1]
  id=unlist(regmatches(x,gregexpr("[[:digit:]]+", x)[1]))[2]
  activity=unlist(str_split(x, "₩₩_"))[1]
  return(cbind(exp_no, id, activity))
}

temp<-data.frame()
for(i in 1:nrow(ch_pt)){
  temp<-rbind(temp, id_f(ch_pt$d[i]))
}

ch_pt2<-cbind(ch_pt, temp)

# 널값 확인 후 0으로 변경
colSums(is.na(ch_pt2))
ch_pt2[is.na(ch_pt2)] <- 0

## 변화 변수만 학습 ##
# x,y 구분
x = ch_pt2 %>% ungroup %>% select(-d, -exp_no, -id, -activity) %>% data.matrix
y = ch_pt2$activity

# xgboost 모델 학습(10-fold)
set.seed(1004)
ch_pt_model = xgb.cv(data = x,label = as.integer(as.factor(y))-1, num_class = levels(as.factor(y)) %>% length,
                     nfold = 10, nrounds = 500, early_stopping_rounds = 8, booster = 'gbtree',
                     objective = 'multi:softprob', eval_metric = 'mlogloss',
                     verbose = F, prediction = T)
```

```
# 예측 데이터 생성(max.col 사용해서 가장 큰값 추출)
pred_df = ch_pt_model$pred %>% as.data.frame %>%
  mutate(pred = levels(as.factor(y))[max.col(.)] %>% as.factor, actual = as.factor(y))

pred_df %>% select(pred,actual) %>% table

# confusionMatrix 생성
Chpoint_conf<- caret::confusionMatrix(pred_df$pred, pred_df$actual)
Chpoint_conf

# 중요 변수 추출
Chpoint_model = xgboost(data = x,label = as.integer(as.factor(y))-1, num_class = levels(as.factor(y)) %>% length,
                        nfold = 10, nrounds = 500, early_stopping_rounds = 8, booster = 'gbtree',
                        objective = 'multi:softprob', eval_metric = 'mlogloss',
                        verbose = F)

Chpoint_imp<- xgb.importance(model = Chpoint_model)

### 모든 변수를 합하여 학습 진행 ####
Static<- HAR_summary_extend %>% ungroup() %>% select(-exp_no,-id,-activity)
Fourier<- fft_data_summary %>% ungroup() %>% select(-exp_no,-id,-activity)
Peak<-Peak_final2 %>% ungroup() %>% select(-exp_no,-id,-activity)
Chpoint<- ch_pt2 %>% ungroup() %>% select(-exp_no,-id)
final<- merge(Static, Fourier, by="d")
final<- merge(final, Peak, by="d")
final<- merge(final, Chpoint, by="d")
final2<- final %>% ungroup() %>% select(-d)

# 널값 확인 후 0으로 변경
colSums(is.na(final2))
final2[is.na(final2)] <- 0

# x,y 구분
x = final2 %>% ungroup %>% select(-activity) %>% data.matrix
y = final2$activity

# xgboost 모델 학습(10-fold)
set.seed(1004)
all_model = xgb.cv(data = x,label = as.integer(as.factor(y))-1, num_class = levels(as.factor(y)) %>% length,
                   nfold = 10, nrounds = 500, early_stopping_rounds = 8, booster = 'gbtree',
                   objective = 'multi:softprob', eval_metric = 'mlogloss',
                   verbose = F, prediction = T)

# 예측 데이터 생성(max.col 사용해서 가장 큰값 추출)
pred_df = all_model$pred %>% as.data.frame %>%
  mutate(pred = levels(as.factor(y))[max.col(.)] %>% as.factor, actual = as.factor(y))

pred_df %>% select(pred,actual) %>% table
```

```
# confusionMatrix 생성
all_conf<- caret::confusionMatrix(pred_df$pred, pred_df$actual)
all_conf

# accuracy 확인
cat("통계 특징 accuracy는", round(Static_conf$overall[[1]],2),"입니다.")
cat("피크 특징 accuracy는", round(Peak_conf$overall[[1]],2),"입니다.")
cat("푸리에 변환 특징 accuracy는", round(Fourier_conf$overall[[1]],2),"입니다.")
cat("변화 특징 accuracy는", round(Chpoint_conf$overall[[1]],2),"입니다.")
cat("총 accuracy는", round(all_conf$overall[[1]],2),"입니다.")

## 가장 높은 정확도를 나타낸 통계특징과 푸리에 변환 특징 변수들만 사용하여 예측 ##
Static<- HAR_summary_extend %>% ungroup() %>% select(-exp_no,-id,-activity)
Fourier<- fft_data_summary %>% ungroup() %>% select(-exp_no,-id)
Sta_Fo<- merge(Static, Fourier, by = 'd')
Sta_Fo<- Sta_Fo %>% ungroup() %>% select(-d)

# 널값 확인 후 0으로 변경
colSums(is.na(Sta_Fo))
Sta_Fo[is.na(Sta_Fo)] <- 0

# x,y 구분
x = Sta_Fo %>% ungroup %>% select(-activity) %>% data.matrix
y = Sta_Fo$activity

# xgboost 모델 학습(10-fold)
set.seed(1004)
Sta_Fo_model = xgb.cv(data = x,label = as.integer(as.factor(y))-1, num_class = levels(as.factor(y)) %>% length,
                      nfold = 10, nrounds = 500, early_stopping_rounds = 8, booster = 'gbtree',
                      objective = 'multi:softprob', eval_metric = 'mlogloss',
                      verbose = F, prediction = T)

# 예측 데이터 생성(max.col 사용해서 가장 큰값 추출)
pred_df = Sta_Fo_model$pred %>% as.data.frame %>%
  mutate(pred = levels(as.factor(y))[max.col(.)] %>% as.factor, actual = as.factor(y))

pred_df %>% select(pred,actual) %>% table

# confusionMatrix 생성
Sta_Fo_conf<- caret::confusionMatrix(pred_df$pred, pred_df$actual)
Sta_Fo_conf

## 각 특징별로 상위 변수들만 추출하여 예측 ##
xgb.ggplot.importance(Static_imp[1:30])
xgb.ggplot.importance(Peak_imp[1:30])
xgb.ggplot.importance(Fourier_imp[1:30])
xgb.ggplot.importance(Chpoint_imp[1:29])
```

```
Staitc_imp_df<- cbind(d = HAR_summary_extend$d,HAR_summary_extend[,Static_imp[1:30]$Feature])
Peak_imp_df<- cbind(d = Peak_final2$d, Peak_final2[,Peak_imp[1:30]$Feature])
Fourier_imp_df<- cbind(d = fft_data_summary$d,fft_data_summary[,Fourier_imp[1:30]$Feature])
Chpoint_imp_df<- cbind(d = ch_pt2$d, activity = ch_pt2$activity, ch_pt2[,Chpoint_imp[1:29]$Feature])

SP_df<- merge(Staitc_imp_df, Peak_imp_df, by = 'd')
SPF_df<- merge(SP_df, Fourier_imp_df, by = 'd')
import_df<-   merge(SPF_df, Chpoint_imp_df, by = 'd')


# x,y 구분
x = import_df %>% ungroup %>% select(-d, -activity) %>% data.matrix
y = import_df$activity

# xgboost 모델 학습(10-fold)
set.seed(1004)
import_model = xgb.cv(data = x,label = as.integer(as.factor(y))-1, num_class = levels(as.factor(y)) %>% length,
                      nfold = 10, nrounds = 500, early_stopping_rounds = 8, booster = 'gbtree',
                      objective = 'multi:softprob', eval_metric = 'mlogloss',
                      verbose = F, prediction = T)

# 예측 데이터 생성(max.col 사용해서 가장 큰값 추출)
pred_df = import_model$pred %>% as.data.frame %>%
  mutate(pred = levels(as.factor(y))[max.col(.)] %>% as.factor, actual = as.factor(y))

pred_df %>% select(pred,actual) %>% table

# confusionMatrix 생성
import_conf<- caret::confusionMatrix(pred_df$pred, pred_df$actual)
import_conf
```