MVA with Subwaydata for Cluster Analysis

KwangMin Son
Department of Statistics, SKKU

November 25, 2018

Need Packages

```r
library(data.table)
library(dplyr)
library(tidyr)
library(stringr)
library(ggplot2)
library(MVA)
```

Loading data(handled from Linux server)

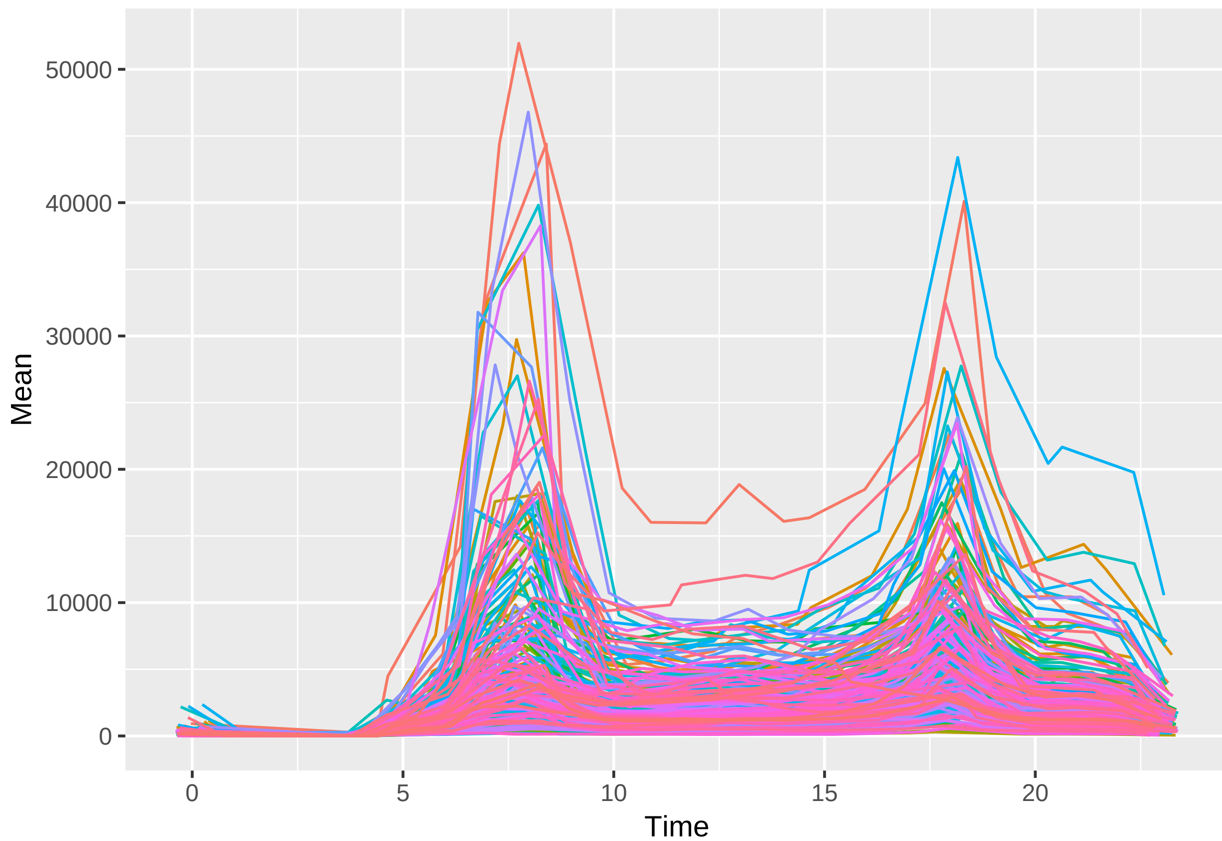
```r
load('subway_dat.RData')
head(dat_weekday_up)
```

```
##            Name Day Time Count
## 1       가락시장  금    0   312
## 2 가산디지털단지  금    0   496
## 3           강남  금    0  6438
## 4       강남구청  금    0   357
## 5           강동  금    0   131
## 6       강동구청  금    0    82
```

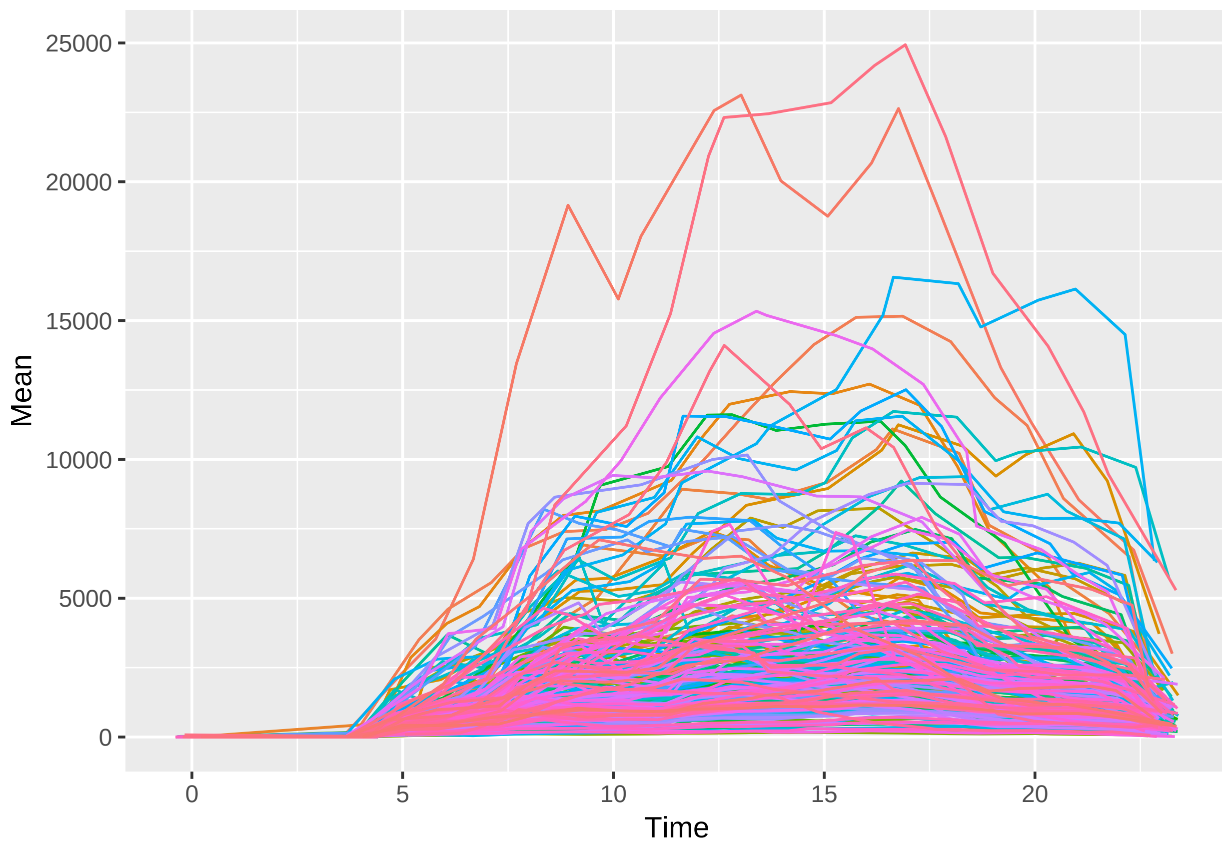
```r
data1 <- fread(file="/home/students/subway/OD_1_8_1804_1.csv")
data2 <- fread(file="/home/students/subway/OD_1_8_1804_2.csv")
data <- rbind(data1, data2)

# 데이터셋 제작과정 예시
data_weekend_up <- data%>%filter(요일%in%c("토","일"))%>%group_by(시간대, 승차역명, 요일)%>%summarise(sum

data_weekday_up <- data%>%filter(요일%in%c("월","화","수","목","금"))%>%group_by(시간대, 승차역명, 요일)%>%

colnames(dat_weekend_up) <- c("Name", "Day","Time","Count")
dat_weekday_up$Name <- str_remove(dat_weekday_up$Name, "$역")
ind1 <- which(str_detect(dat_weekday_up$Name, "[(]"))
ind2 <- str_locate(dat_weekday_up$Name[ind1], "[(]")
dat_weekday_up$Name[ind1] <- str_sub(dat_weekday_up$Name[ind1],1,ind2-1)

dat_weekend_up$Name <- str_remove(dat_weekend_up$Name, "$역")
ind1 <- which(str_detect(dat_weekend_up$Name, "[(]"))
ind2 <- str_locate(dat_weekend_up$Name[ind1], "[(]")
dat_weekend_up$Name[ind1] <- str_sub(dat_weekend_up$Name[ind1],1,ind2-1)
```

calculate mean from weekend/weeday(down)

```
dat_weekend_mean_down <- dat_weekend_down%>%group_by(Name, Time)%>%summarise(Mean=mean(Count))
dat_weekday_mean_down <- dat_weekday_down%>%group_by(Name, Time)%>%summarise(Mean=mean(Count))

### 평일데이터에 대하여 1달 하차 자료의 시간별 평균 분포 확인
ggplot(data= dat_weekday_mean_down, aes(x=Time, y=Mean, color=Name)) + geom_line(position = 'jitter') +
```




```
### 주말데이터에 대하여 1달 하차 자료의 시간별 평균 분포 확인
ggplot(data= dat_weekend_mean_down, aes(x=Time, y=Mean, color=Name)) + geom_line(position = 'jitter') +
```



Making Group through MDS technique(down)

```
dat_weekend_mat_down <- dat_weekend_mean_down%>%spread(Time, Mean)
dat_weekday_mat_down <- dat_weekday_mean_down%>%spread(Time, Mean)
dim(dat_weekday_mat_down)
```

```
## [1] 241  24
```

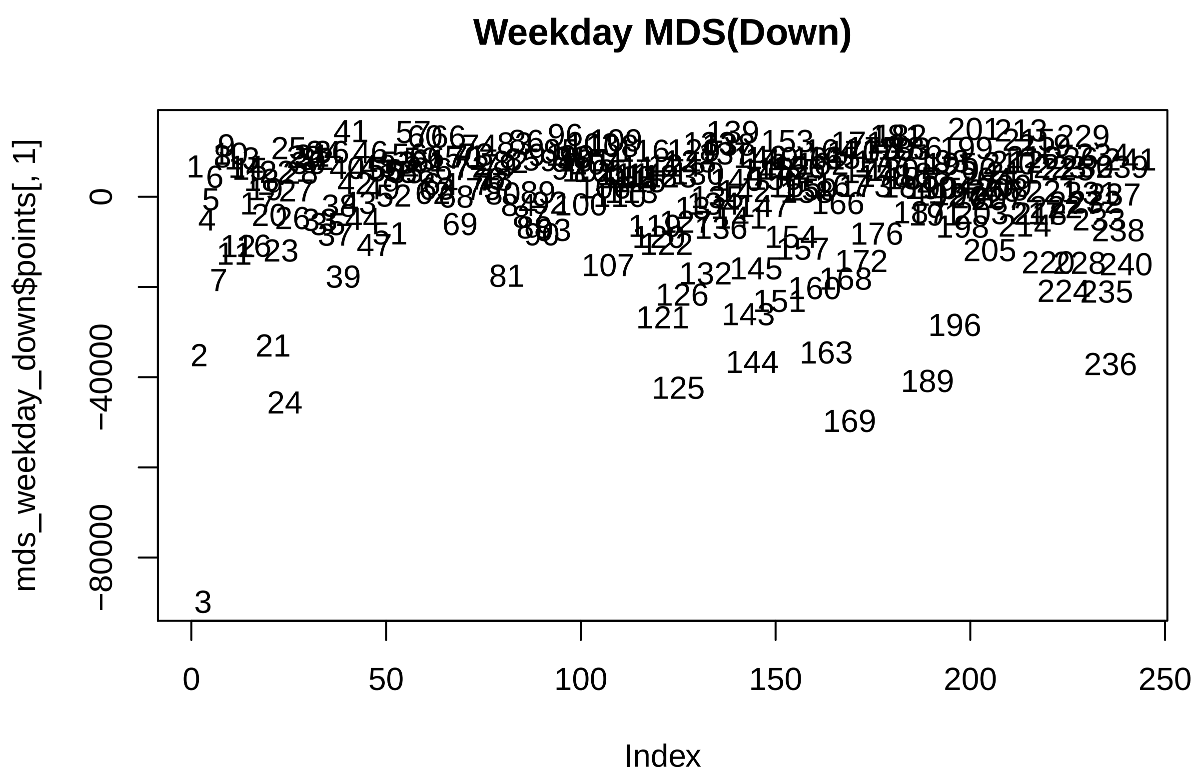
```
# NA 값 => 0으로 처리하였음.
for(i in 1:241){
  dat_weekend_mat_down[i,which(is.na(dat_weekend_mat_down[i,]))] <- 0
  dat_weekday_mat_down[i,which(is.na(dat_weekday_mat_down[i,]))] <- 0
}
colnames(dat_weekend_mat_down) <- c("Name", paste0("time", c(0:2, 4:23)))
colnames(dat_weekday_mat_down) <- c("Name", paste0("time", c(0:2, 4:23)))

dist_weekend_down <- dist(dat_weekend_mat_down[,-1])
dist_weekday_down <- dist(dat_weekday_mat_down[,-1])

#평일 자료에 대한 MDS
mds_weekday_down <- cmdscale(dist_weekday_down, k = 10, eig = TRUE)
mds_weekday_down_eig <- mds_weekday_down$eig
head(cumsum(abs(mds_weekday_down_eig)) / sum(abs(mds_weekday_down_eig)))
```

```
## [1] 0.6821672 0.9607950 0.9823952 0.9904063 0.9942610 0.9969030
```

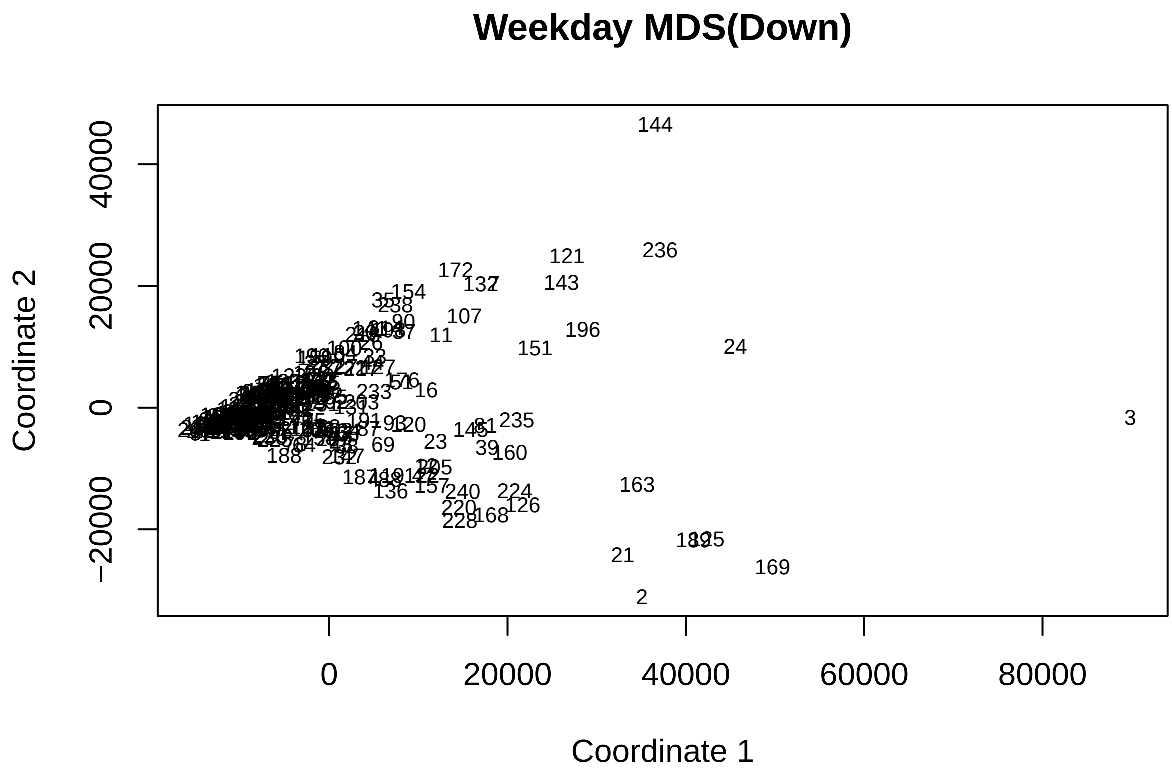
```r
head(cumsum((mds_weekday_down_eig)^2) / sum((mds_weekday_down_eig)^2))
```

```
## [1] 0.8561508 0.9989801 0.9998385 0.9999566 0.9999839 0.9999967
```

```r
#처음 1개의 차원으로 충분 확인 후 1/2차원 표현
plot(mds_weekday_down$points[,1], type = "n", main = "Weekday MDS(Down)")
text(mds_weekday_down$points[,1], labels(sort(unique(dat_weekday_down$Name))))
```




```r
plot(mds_weekday_down$points[,1] * (-1), mds_weekday_down$points[,2] * (-1),
     type = "n", xlab = "Coordinate 1", ylab = "Coordinate 2", main = "Weekday MDS(Down)")
text(mds_weekday_down$points[,1] *(-1), mds_weekday_down$points[,2] * (-1),
     labels(sort(unique(dat_weekday_down$Name))), cex = 0.7)
```



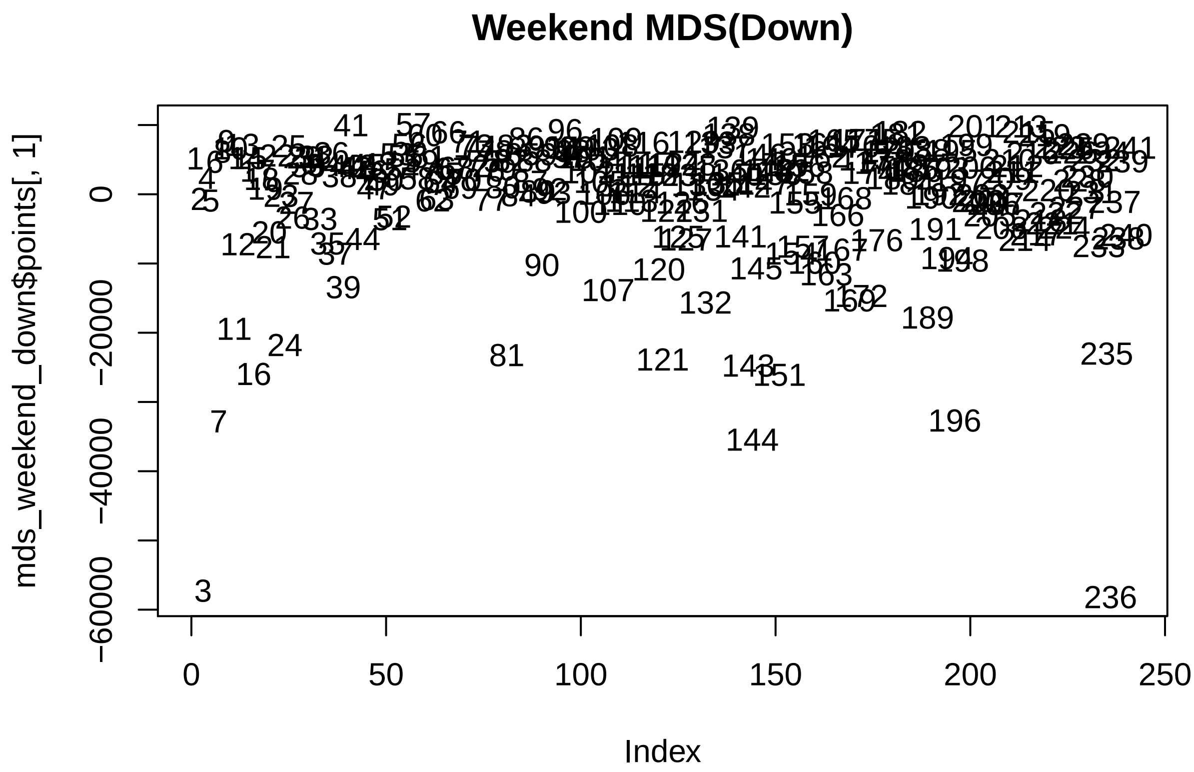

```
### 평일의 경우 근무지가 집중된 회사밀집지역에 대하여 데이터가 구분되는 것을 확인.

#주말 자료에 대한 MDS
mds_weekend_down <- cmdscale(dist_weekend_down, k = 10, eig = TRUE)
mds_weekend_down_eig <- mds_weekend_down$eig
head(cumsum(abs(mds_weekend_down_eig)) / sum(abs(mds_weekend_down_eig)))
```

```
## [1] 0.8895788 0.9718687 0.9869709 0.9917341 0.9946659 0.9958586
```

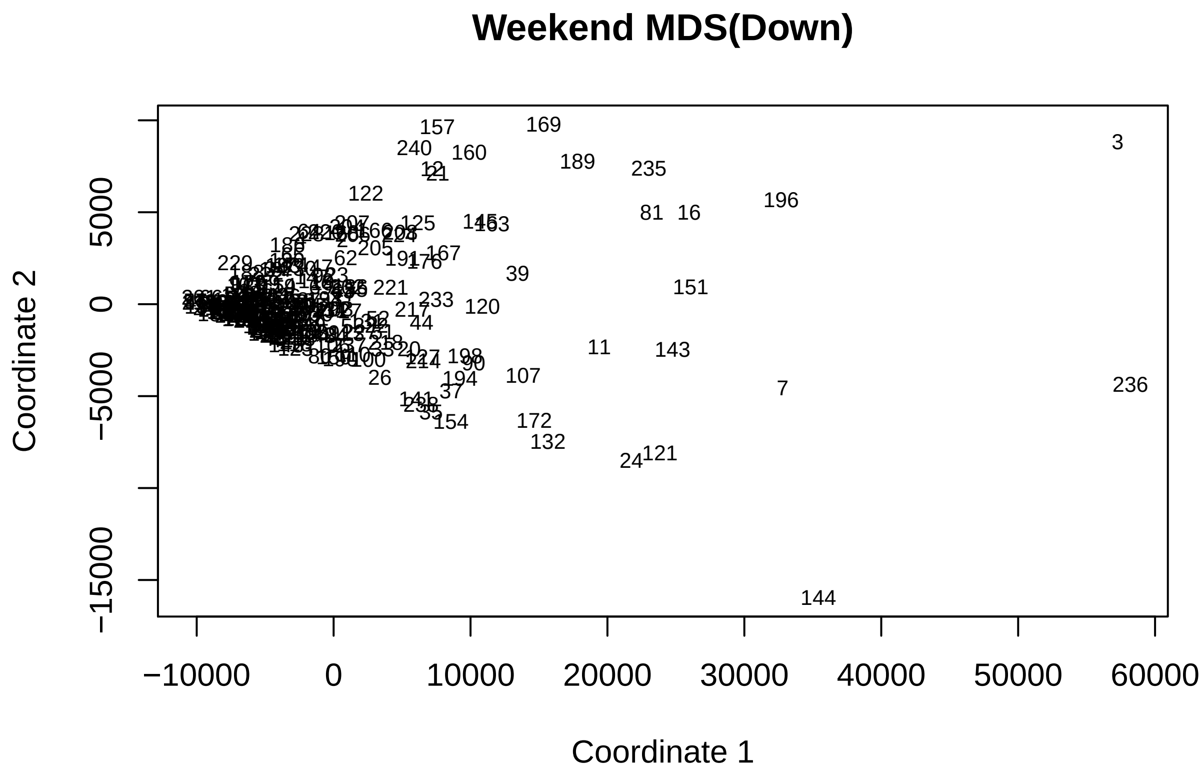
```
head(cumsum((mds_weekend_down_eig)^2) / sum((mds_weekend_down_eig)^2))
```

```
## [1] 0.9911885 0.9996701 0.9999558 0.9999842 0.9999950 0.9999968
```

```
#처음 2개의 차원으로 충분 확인 후 1/2차원 표현
plot(mds_weekend_down$points[,1], type = "n", main = "Weekend MDS(Down)")
text(mds_weekend_down$points[,1], labels(sort(unique(dat_weekday_down$Name))))
```



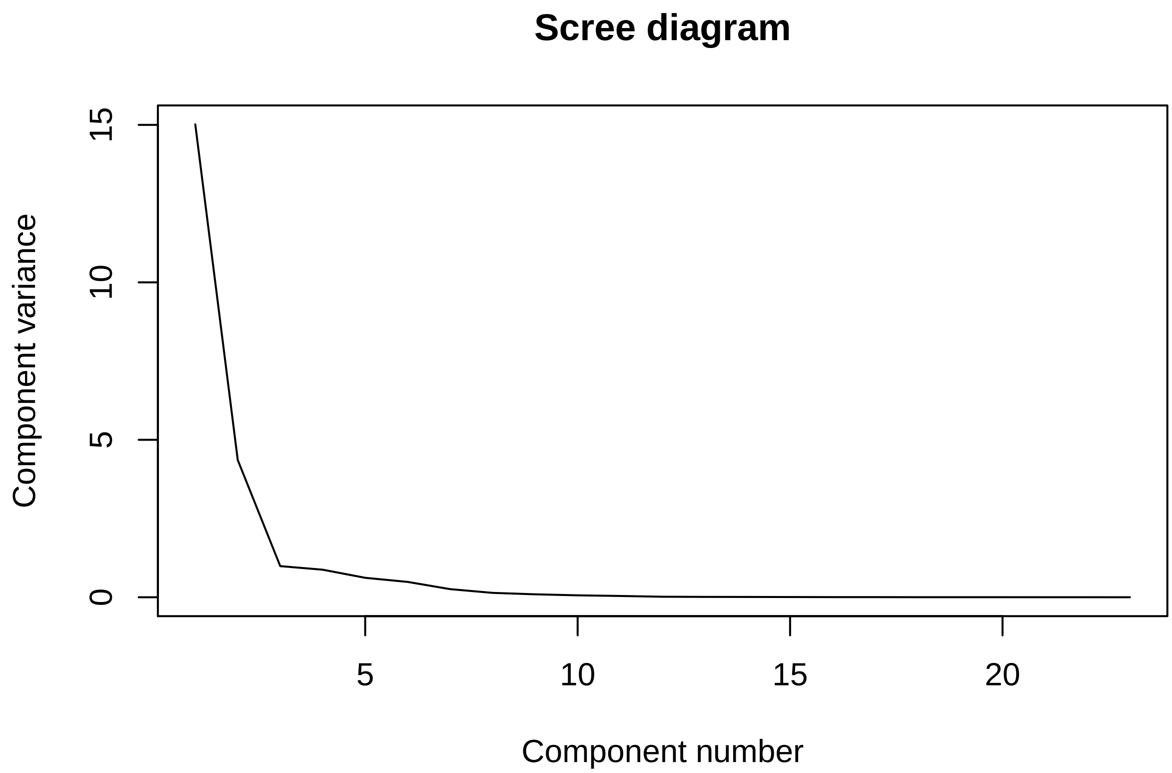

```r
plot(mds_weekend_down$points[,1] * (-1), mds_weekend_down$points[,2] * (-1),
     type = "n", xlab = "Coordinate 1", ylab = "Coordinate 2", main = "Weekend MDS(Down)")
text(mds_weekend_down$points[,1] *(-1), mds_weekend_down$points[,2] * (-1),
     labels(sort(unique(dat_weekday_down$Name))), cex = 0.7)
```




```
### 주말의 경우 유흥지가 집중된 번화가에 대하여 데이터가 구분되는 것을 확인.
```

Making Group through PCA technique(down)

```
pca_weekday_down <- prcomp(dat_weekday_mat_down[,-1], scale = TRUE)
plot(pca_weekday_down$sdev^2, xlab = "Component number",
     ylab = "Component variance", type = "l", main = "Scree diagram")
```




```r
cumsum(pca_weekday_down$sdev^2)/sum(pca_weekday_down$sdev^2)
```

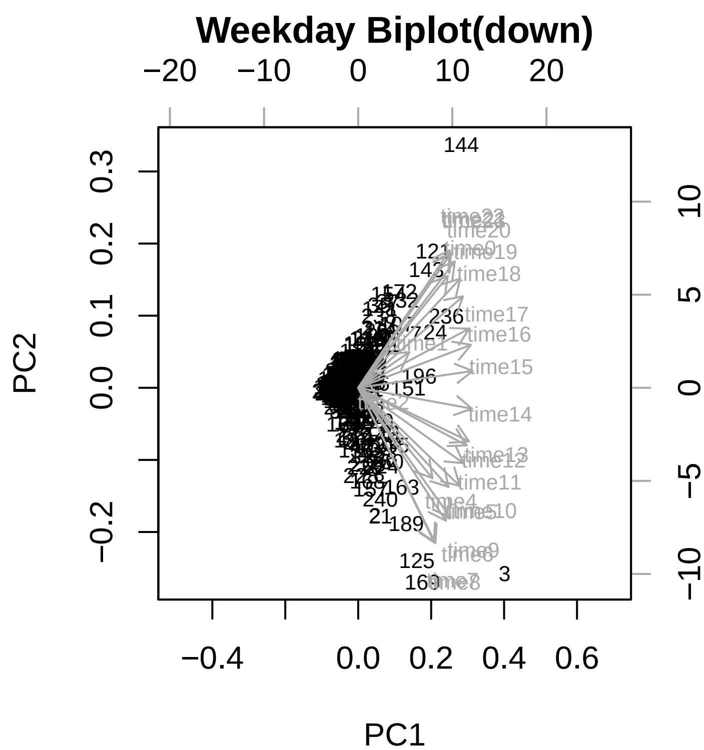
```
##  [1] 0.6528823 0.8423519 0.8853276 0.9233688 0.9502273 0.9714260 0.9826341
##  [8] 0.9886994 0.9927635 0.9954489 0.9972319 0.9979369 0.9984485 0.9988668
## [15] 0.9991735 0.9993837 0.9995771 0.9996845 0.9997844 0.9998708 0.9999225
## [22] 0.9999697 1.0000000
```

```r
### weekday의 경우 PC 2개로 총 변동에 84.24%를 설명하므로 충분해보임
pca_weekday_down$rotation[,1:2]
```

```
##               PC1         PC2
## time0  0.19774524  0.23160783
## time1  0.11275116  0.07400659
## time2  0.04471205 -0.02330175
## time4  0.16403512 -0.18730886
## time5  0.19936291 -0.20544332
## time6  0.19327307 -0.27520544
## time7  0.16684445 -0.31824332
## time8  0.17041143 -0.32204505
## time9  0.20370705 -0.26888577
## time10 0.22391376 -0.20321138
## time11 0.23244534 -0.15632785
## time12 0.23936905 -0.11934752
## time13 0.24425420 -0.11096568
## time14 0.25101570 -0.04320627
## time15 0.25265194  0.03490176
## time16 0.24898016  0.08910353
```

```
## time17 0.24473133  0.12261415
## time18 0.23120367  0.18956723
## time19 0.22491671  0.22591598
## time20 0.21314142  0.26176080
## time21 0.20500110  0.27944241
## time22 0.20197922  0.28503486
## time23 0.20410362  0.27720463
```

```
### PC1 : 전반적인 하차객수, PC2 : 오전시간 대비 오후시간대의 하차객수(특히 출근시간대 6,7,8,9 높은 값을 가짐)
tmp <- dat_weekday_mat_down[,-1]
biplot(prcomp(tmp, scale = TRUE), col = c("black", "darkgray"), xlim =
c(-0.5, 0.7), cex = 0.7, main = "Weekday Biplot(down)")
```




```
pca_weekend_down <- prcomp(dat_weekend_mat_down[,-1], scale = TRUE)
cumsum(pca_weekend_down$sdev^2)/sum(pca_weekend_down$sdev^2)
```

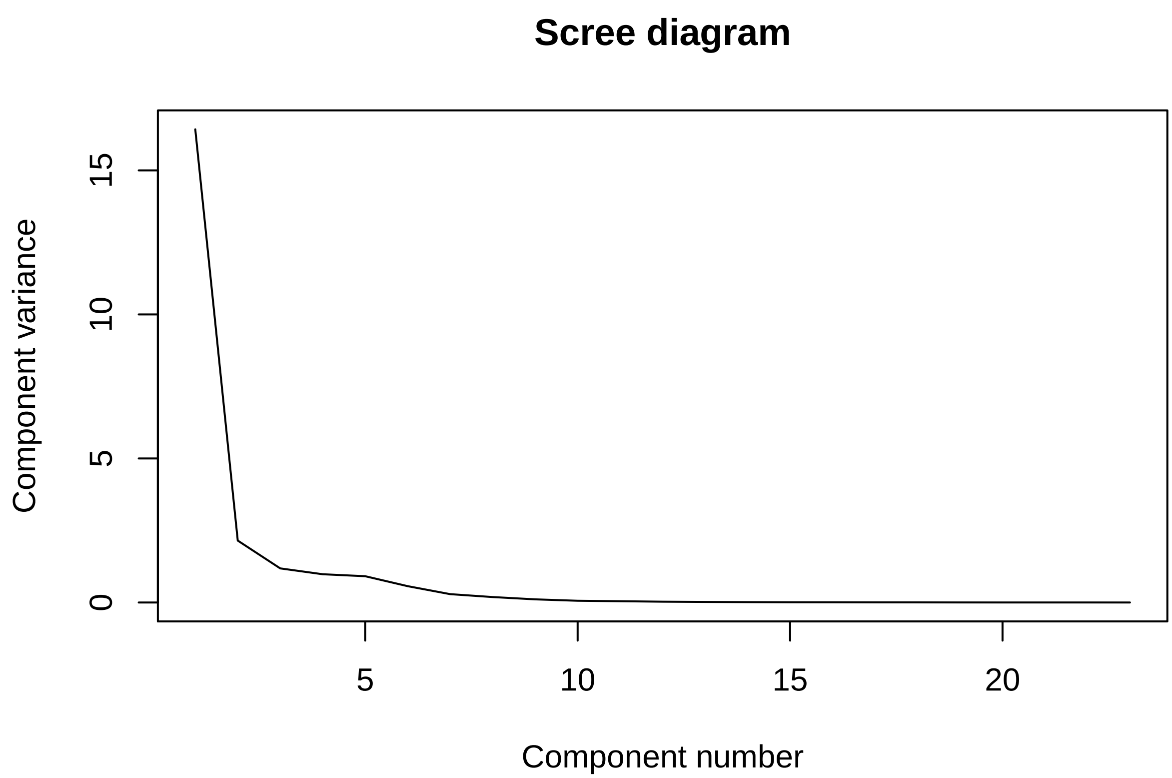
```
##  [1] 0.7141666 0.8075871 0.8590660 0.9017137 0.9413710 0.9660051 0.9785947
##  [8] 0.9868261 0.9915711 0.9942065 0.9961241 0.9973327 0.9981738 0.9987387
## [15] 0.9991174 0.9994312 0.9996381 0.9997928 0.9998696 0.9999186 0.9999589
## [22] 0.9999859 1.0000000
```

```
### weekday의 경우 PC 2개로 총 변동에 80%를 설명하므로 충분해보임
pca_weekend_down$rotation[,1:2]
```

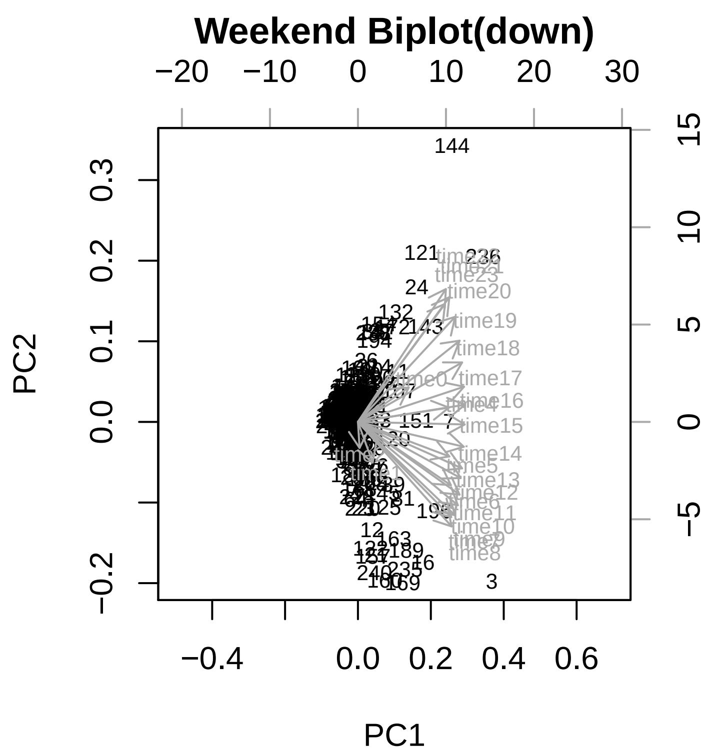
```
##               PC1          PC2
## time0  0.114663252  0.097062076
## time1  0.033292839 -0.114752947
## time2  0.003634711 -0.073362709
```

```
## time4  0.205256254  0.040342663
## time5  0.206304508 -0.097419601
## time6  0.209757335 -0.178485049
## time7  0.210542994 -0.268785930
## time8  0.211440611 -0.294890890
## time9  0.219151400 -0.263434115
## time10 0.225527722 -0.235149624
## time11 0.229301647 -0.205117676
## time12 0.231895149 -0.158423241
## time13 0.235096788 -0.131019893
## time14 0.238893409 -0.070085417
## time15 0.240867844 -0.007114838
## time16 0.241750270  0.048552156
## time17 0.239513635  0.099305725
## time18 0.235068681  0.167697865
## time19 0.229532820  0.229257550
## time20 0.219268856  0.296347424
## time21 0.206336294  0.351842301
## time22 0.198479789  0.374805259
## time23 0.196475285  0.332956320
```

```
### PC1 : 전반적인 하차객수, PC2 : 오전시간 대비 오후시간대의 하차객수(특히 오후 7시 이후 유흥지역이 활발히 운영

plot(pca_weekend_down$sdev^2, xlab = "Component number",
     ylab = "Component variance", type = "l", main = "Scree diagram")
```

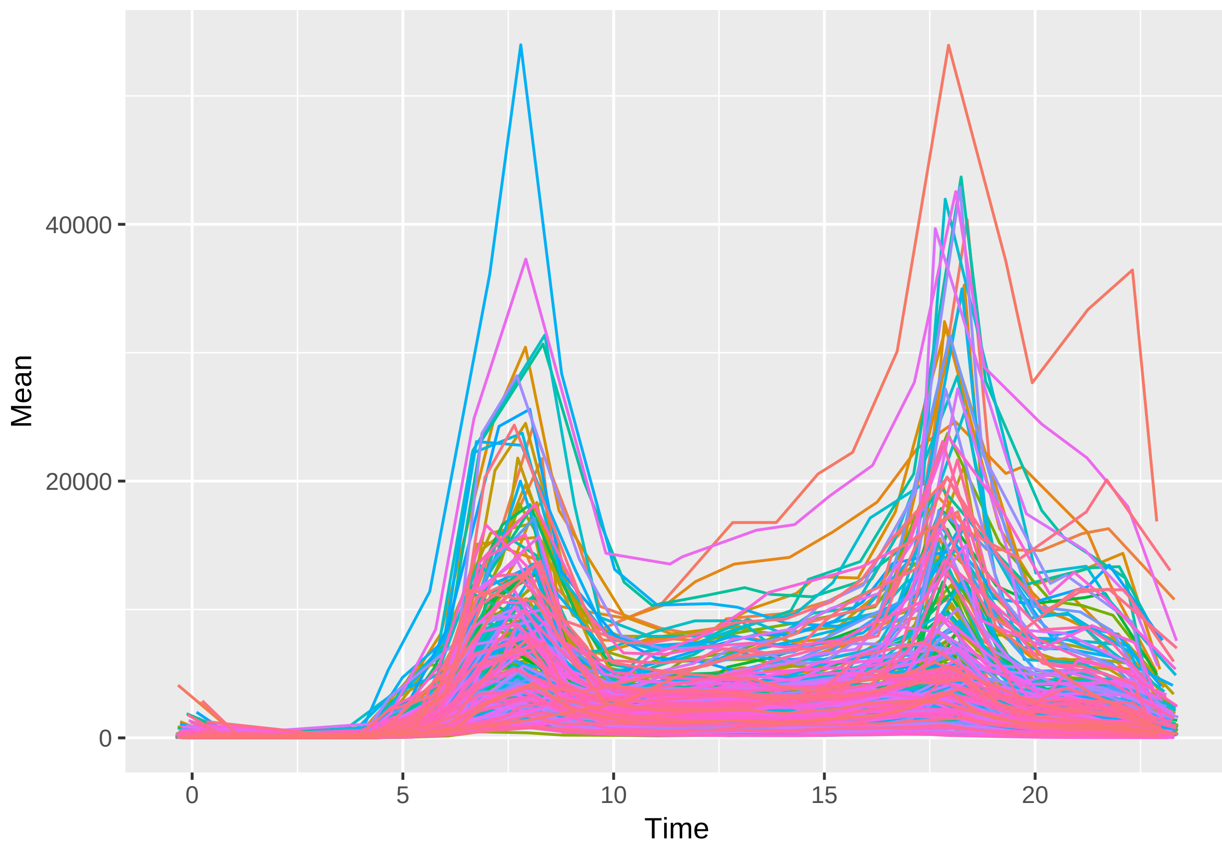


```
tmp2 <- dat_weekend_mat_down[,-1]
biplot(prcomp(tmp2, scale = TRUE), col = c("black", "darkgray"), xlim =
c(-0.5, 0.7), cex = 0.7, main = "Weekend Biplot(down)")
```

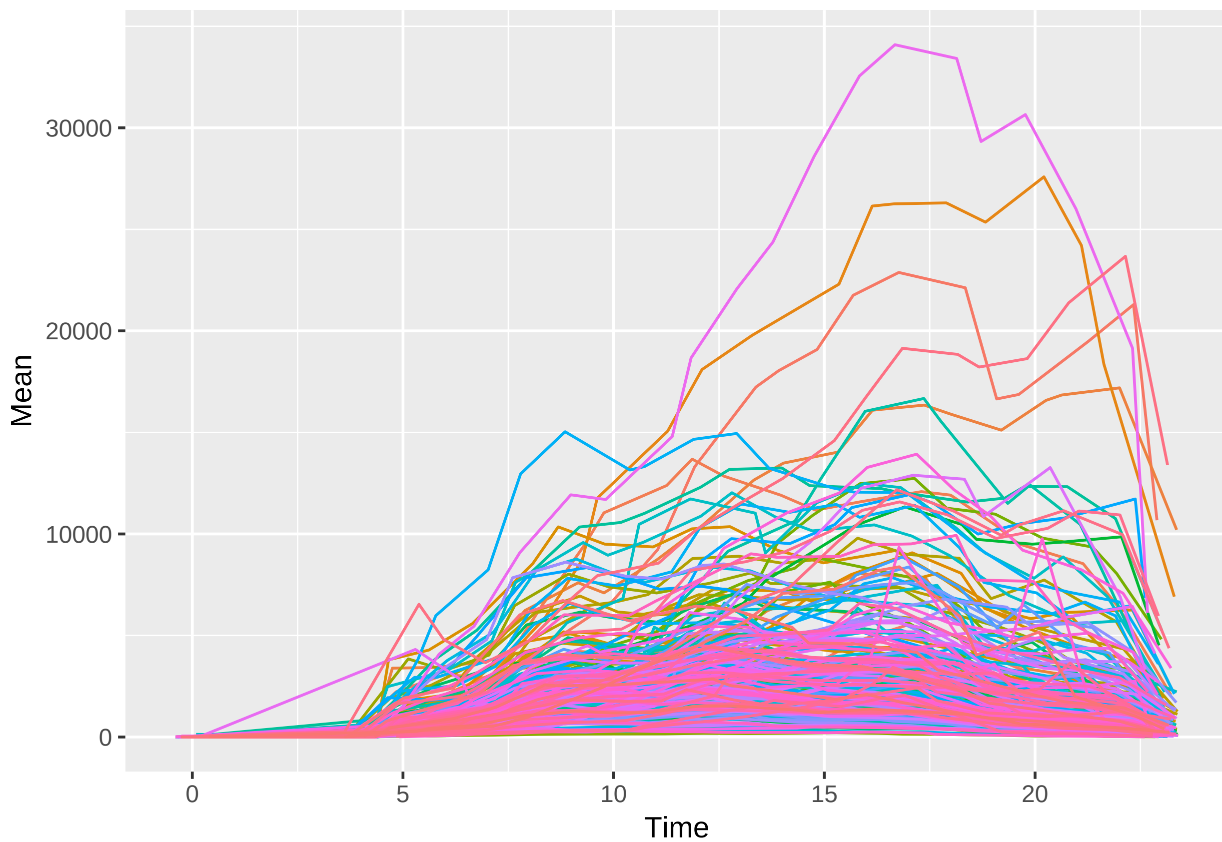


calculate mean from weekend/weeday(up)

```
dat_weekend_mean_up <- dat_weekend_up%>%group_by(Name, Time)%>%summarise(Mean=mean(Count))
dat_weekday_mean_up <- dat_weekday_up%>%group_by(Name, Time)%>%summarise(Mean=mean(Count))

### 평일데이터에 대하여 1달 하차 자료의 시간별 평균 분포 확인
ggplot(data= dat_weekday_mean_up, aes(x=Time, y=Mean, color=Name)) + geom_line(position = 'jitter') +
```




```
### 주말데이터에 대하여 1달 하차 자료의 시간별 평균 분포 확인
ggplot(data= dat_weekend_mean_up, aes(x=Time, y=Mean, color=Name)) + geom_line(position = 'jitter') +  t
```



Making Group through MDS technique(up)

```
dat_weekend_mat_up <- dat_weekend_mean_up%>%spread(Time, Mean)
dat_weekday_mat_up <- dat_weekday_mean_up%>%spread(Time, Mean)
dim(dat_weekday_mat_up)
```

```
## [1] 242  24
```

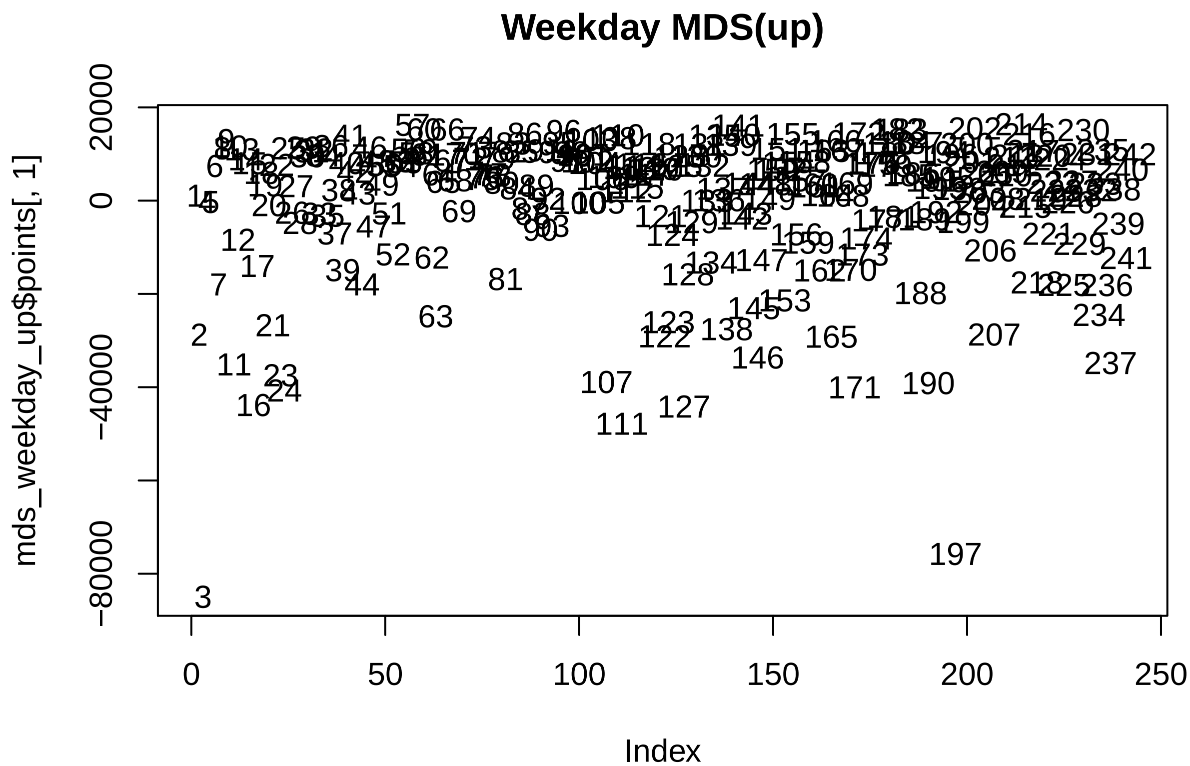
```
# NA 값 => 0으로 처리하였음.
for(i in 1:242){
  dat_weekend_mat_up[i,which(is.na(dat_weekend_mat_up[i,]))] <- 0
  dat_weekday_mat_up[i,which(is.na(dat_weekday_mat_up[i,]))] <- 0
}
colnames(dat_weekend_mat_up) <- c("Name", paste0("time", c(0:2, 4:23)))
colnames(dat_weekday_mat_up) <- c("Name", paste0("time", c(0:2, 4:23)))
dist_weekend_up <- dist(dat_weekend_mat_up[,-1])
dist_weekday_up <- dist(dat_weekday_mat_up[,-1])

#평일 자료에 대한 MDS
mds_weekday_up <- cmdscale(dist_weekday_up, k = 10, eig = TRUE)
mds_weekday_up_eig <- mds_weekday_up$eig
head(cumsum(abs(mds_weekday_up_eig)) / sum(abs(mds_weekday_up_eig)))
```

```
## [1] 0.6853618 0.9486669 0.9769549 0.9880480 0.9922885 0.9947124
```

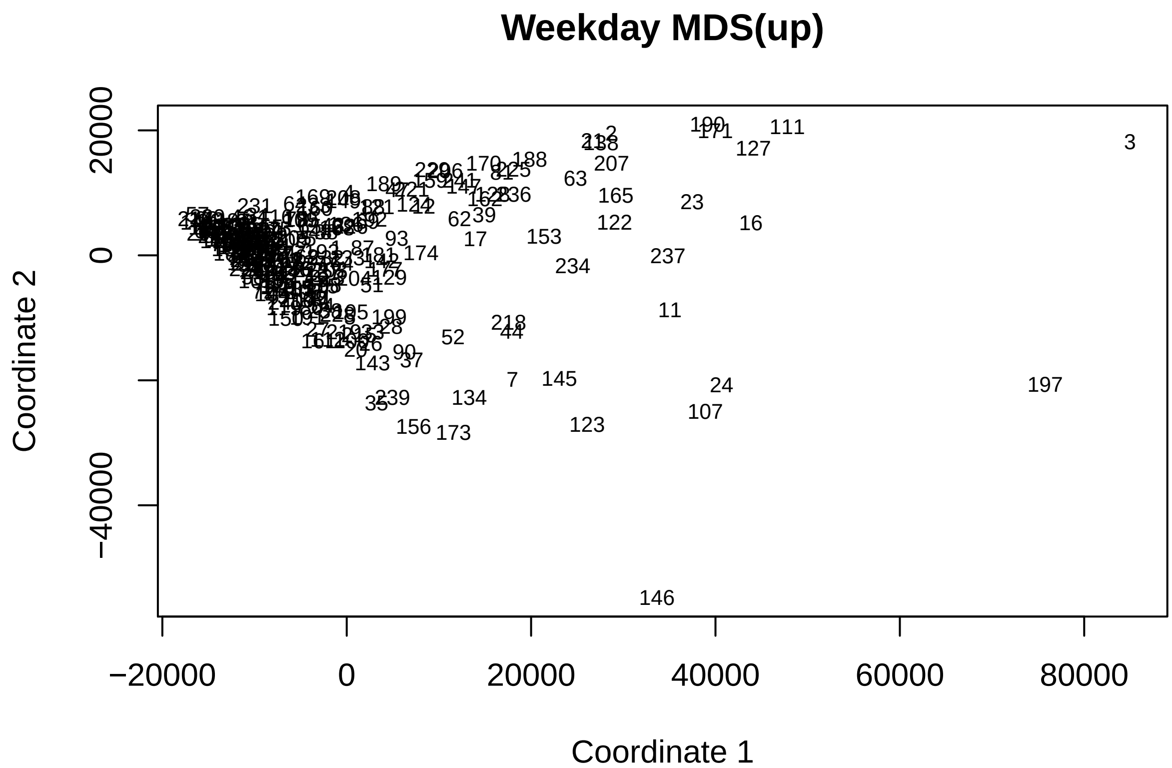
```r
head(cumsum((mds_weekday_up_eig)^2) / sum((mds_weekday_up_eig)^2))
```

```
## [1] 0.8698501 0.9982377 0.9997196 0.9999475 0.9999808 0.9999916
```

```r
#처음 2개의 차원으로 충분 확인 후 1/2차원 표현
plot(mds_weekday_up$points[,1], type = "n", main = "Weekday MDS(up)")
text(mds_weekday_up$points[,1], labels(sort(unique(dat_weekday_up$Name))))
```




```r
plot(mds_weekday_up$points[,1] * (-1), mds_weekday_up$points[,2] * (-1),
     type = "n", xlab = "Coordinate 1", ylab = "Coordinate 2", main = "Weekday MDS(up)")
text(mds_weekday_up$points[,1] *(-1), mds_weekday_up$points[,2] * (-1),
     labels(sort(unique(dat_weekday_up$Name))), cex = 0.7)
```



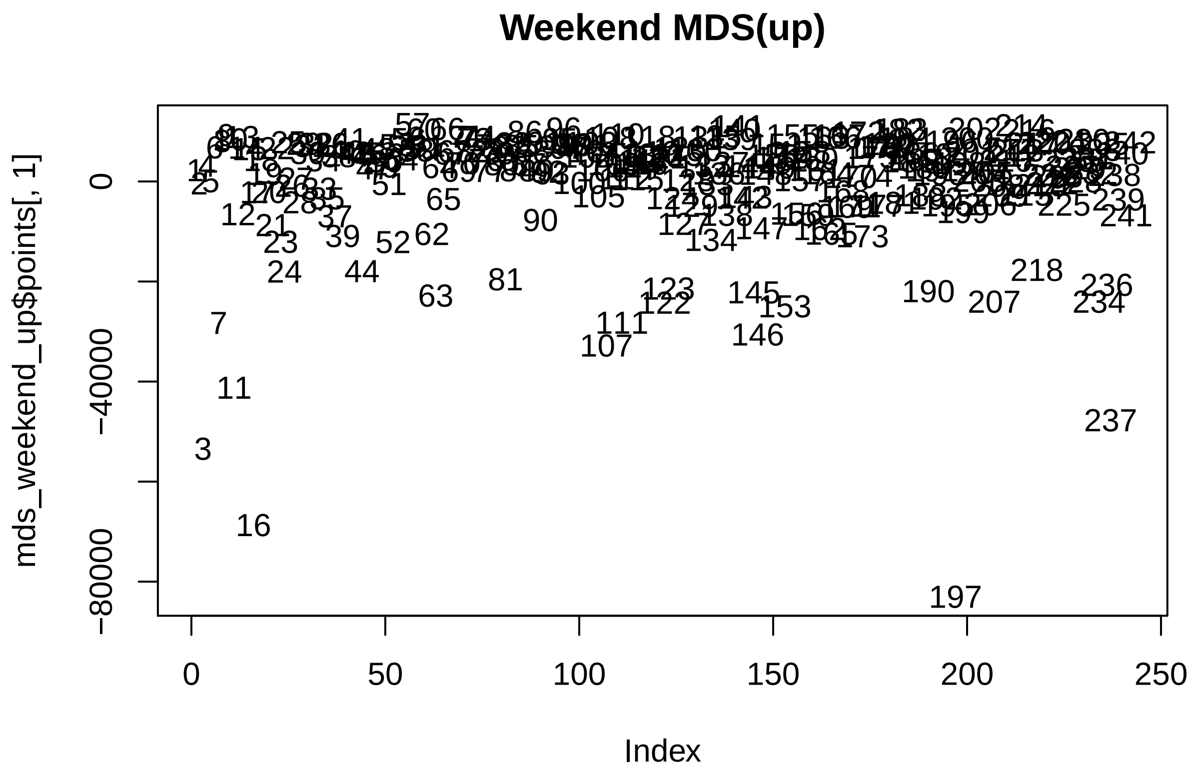

```
### 평일의 경우 근무지가 집중된 회사밀집지역에 대하여 데이터가 구분되는 것을 확인.

#주말 자료에 대한 MDS
mds_weekend_up <- cmdscale(dist_weekend_up, k = 10, eig = TRUE)
mds_weekend_up_eig <- mds_weekend_up$eig
head(cumsum(abs(mds_weekend_up_eig)) / sum(abs(mds_weekend_up_eig)))
```

```
## [1] 0.8981067 0.9696176 0.9864360 0.9918528 0.9950296 0.9962576
```

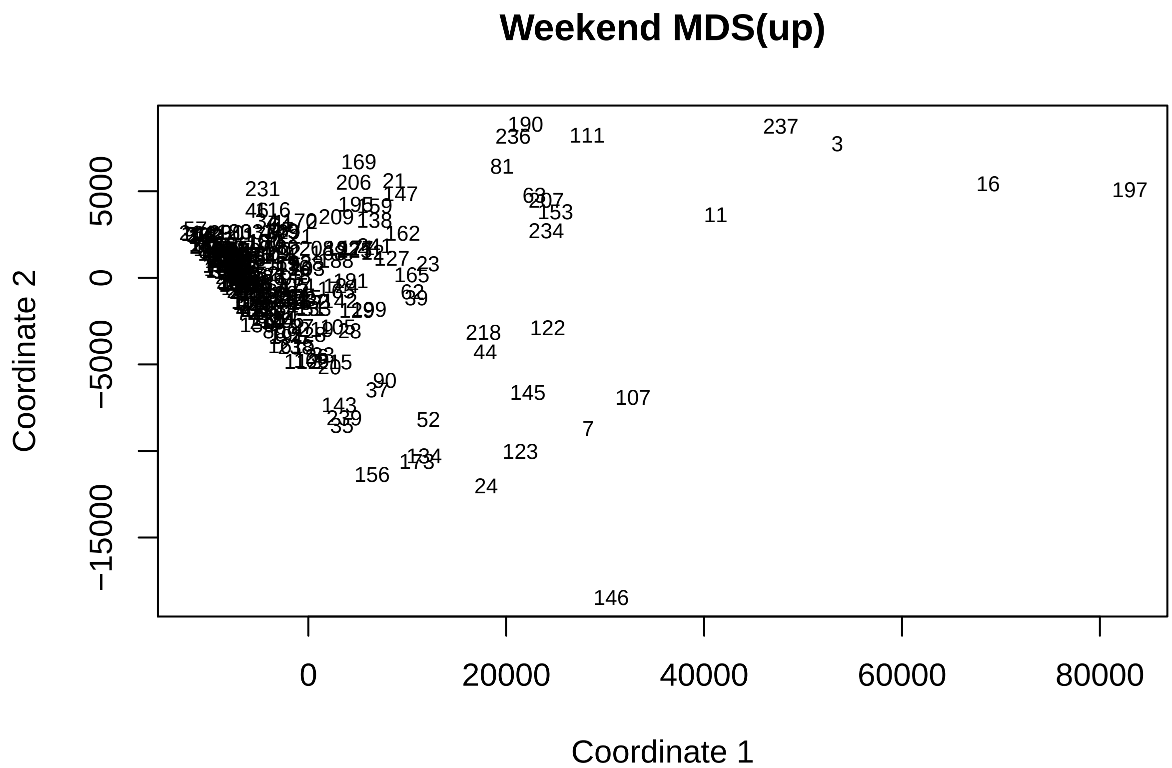
```
head(cumsum((mds_weekend_up_eig)^2) / sum((mds_weekend_up_eig)^2))
```

```
## [1] 0.9933014 0.9995989 0.9999473 0.9999834 0.9999958 0.9999977
```

```
#처음 1개의 차원으로 충분 확인 후 1/2차원 표현
plot(mds_weekend_up$points[,1], type = "n", main = "Weekend MDS(up)")
text(mds_weekend_up$points[,1], labels(sort(unique(dat_weekday_up$Name))))
```



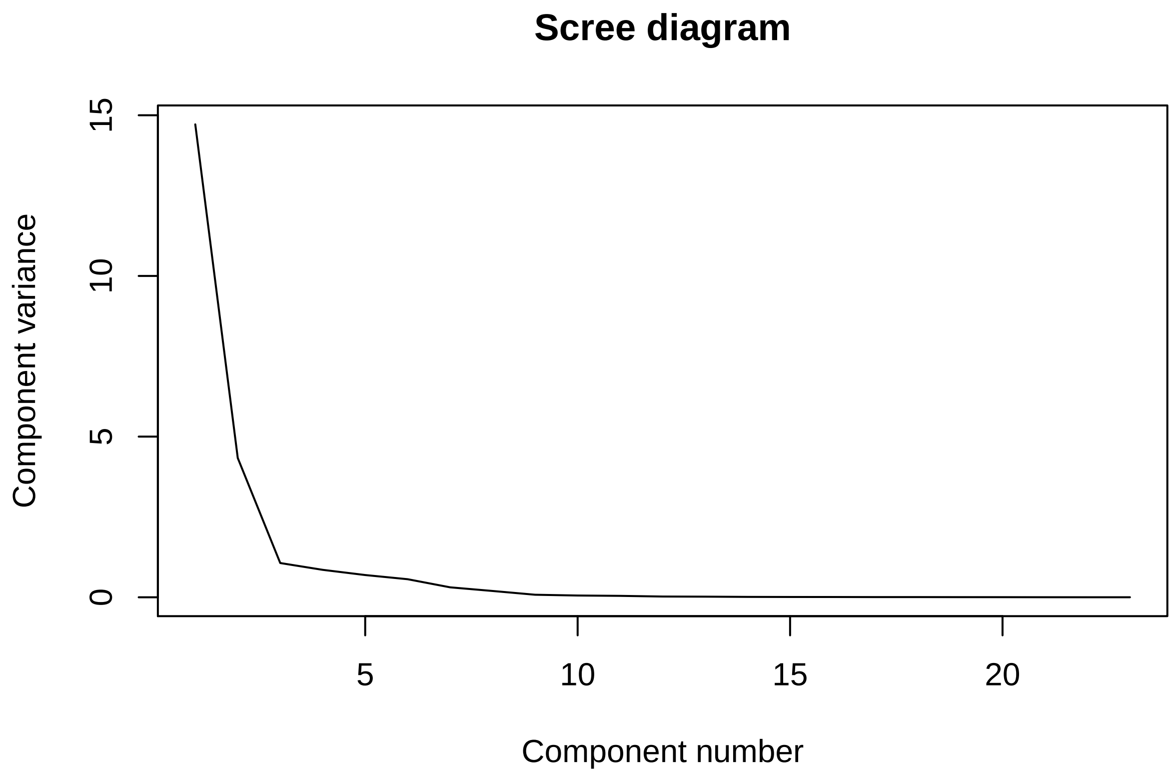

```r
plot(mds_weekend_up$points[,1] * (-1), mds_weekend_up$points[,2] * (-1),
     type = "n", xlab = "Coordinate 1", ylab = "Coordinate 2", main = "Weekend MDS(up)")
text(mds_weekend_up$points[,1] *(-1), mds_weekend_up$points[,2] * (-1),
     labels(sort(unique(dat_weekday_up$Name))), cex = 0.7)
```




```
### 주말의 경우 유흥지가 집중된 번화가에 대하여 데이터가 구분되는 것을 확인.
### 또는 잠실(야구경기장), 고속터미널(시외고속버스터미널)에 대한 승차건수가 많음
```

Making Group through PCA technique(up)

```r
pca_weekday_up <- prcomp(dat_weekday_mat_up[,-1], scale = TRUE)
plot(pca_weekday_up$sdev^2, xlab = "Component number",
     ylab = "Component variance", type = "l", main = "Scree diagram")
```




```
cumsum(pca_weekday_up$sdev^2)/sum(pca_weekday_up$sdev^2)
```

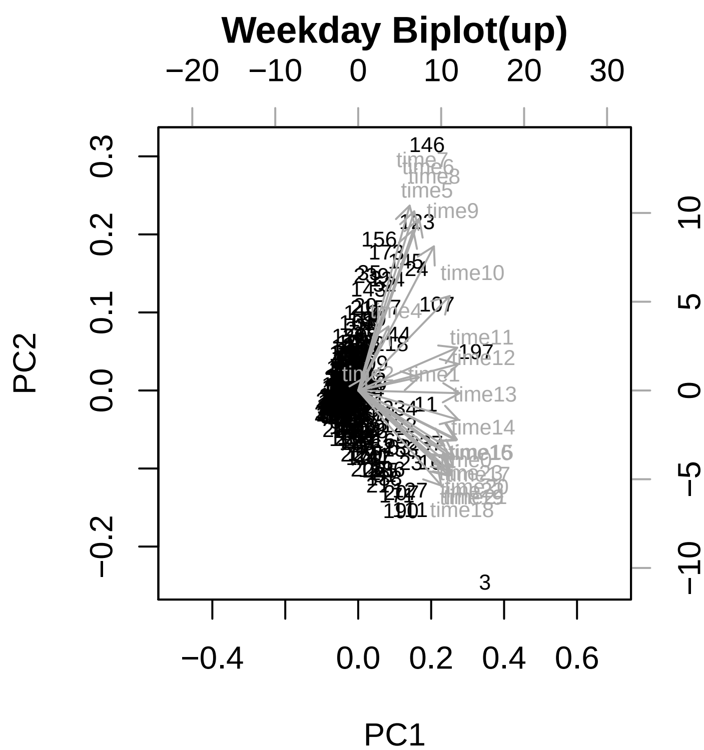
```
##  [1] 0.6398720 0.8283027 0.8746185 0.9117375 0.9418227 0.9662476 0.9797222
##  [8] 0.9882272 0.9916446 0.9940274 0.9959454 0.9969511 0.9977677 0.9982736
## [15] 0.9986848 0.9990304 0.9992742 0.9995096 0.9996800 0.9998218 0.9998987
## [22] 0.9999519 1.0000000
```

```
### weekday의 경우 PC 2개로 총 변동에 82.83%를 설명하므로 충분해보임
pca_weekday_up$rotation[,1:2]
```

```
##                PC1          PC2
## time0  0.21759164 -0.119859939
## time1  0.15340403  0.028757030
## time2  0.02004383  0.029651208
## time4  0.07647313  0.139285869
## time5  0.13908099  0.348786142
## time6  0.14155355  0.389605299
## time7  0.12994731  0.401966713
## time8  0.15391880  0.373381954
## time9  0.19118755  0.313378005
## time10 0.23107577  0.205427989
## time11 0.25005538  0.092986483
## time12 0.25290820  0.057502684
## time13 0.25617043 -0.005937642
## time14 0.25276751 -0.063440772
## time15 0.24920090 -0.106779599
## time16 0.24706275 -0.108331394
```

```
## time17 0.24278083 -0.144892229
## time18 0.20983893 -0.206755290
## time19 0.22947480 -0.185752249
## time20 0.23958875 -0.166902652
## time21 0.23636398 -0.184550500
## time22 0.23054217 -0.173520735
## time23 0.22751754 -0.142352469
```

```
### PC1 : 전반적인 승차객수, PC2 : 오후시간 대비 오전시간대의 승차객수(특히 출근시간대 5,6,7,8,9시에 높은 값을
tmp <- dat_weekday_mat_up[,-1]
biplot(prcomp(tmp, scale = TRUE), col = c("black", "darkgray"), xlim =
c(-0.5, 0.7), cex = 0.7, main = "Weekday Biplot(up)")
```




```
pca_weekend_up <- prcomp(dat_weekend_mat_up[,-1], scale = TRUE)
cumsum(pca_weekend_up$sdev^2)/sum(pca_weekend_up$sdev^2)
```

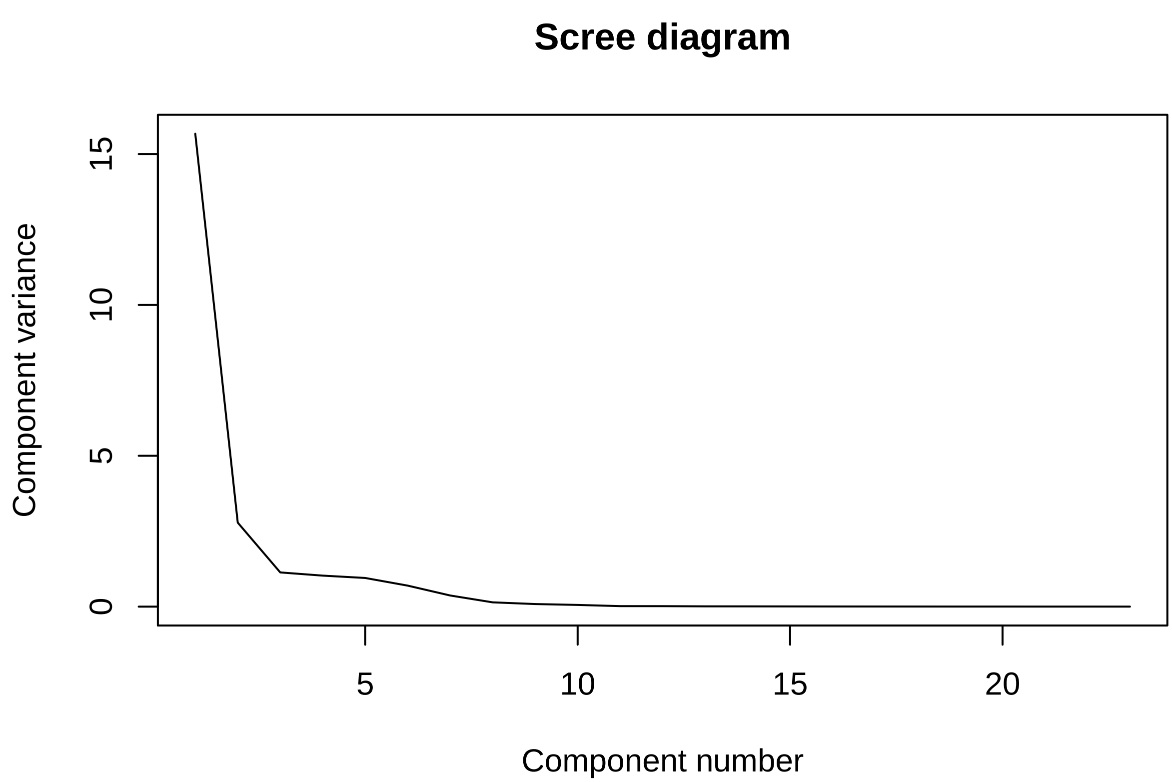
```
##  [1] 0.6814952 0.8025398 0.8518298 0.8965135 0.9378336 0.9681118 0.9841683
##  [8] 0.9903383 0.9940731 0.9965211 0.9972550 0.9979248 0.9983616 0.9987301
## [15] 0.9990464 0.9992608 0.9994359 0.9995773 0.9996955 0.9997915 0.9998776
## [22] 0.9999535 1.0000000
```

```
### weekday의 경우 PC 2개로 총 변동에 80%를 설명하므로 충분해보임
pca_weekend_up$rotation[,1:2]
```

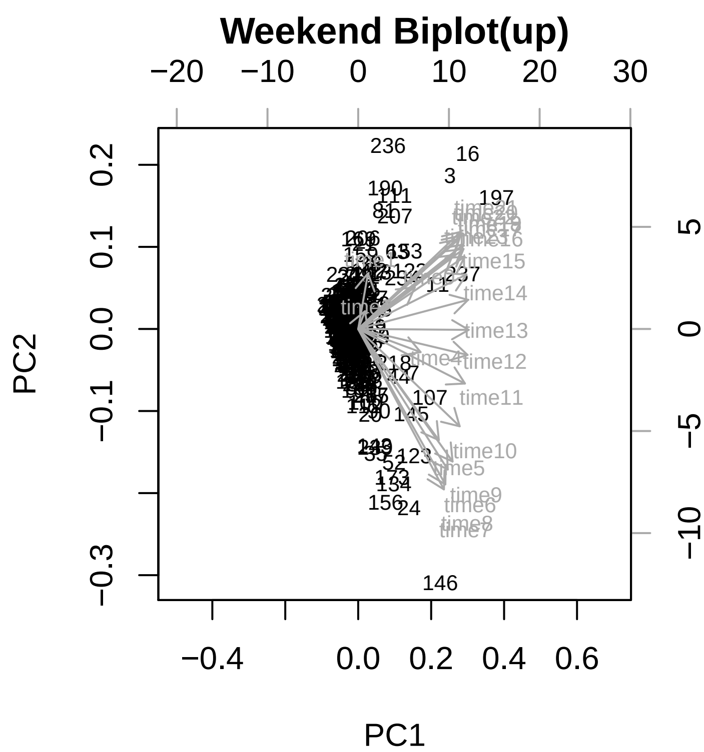
```
##               PC1         PC2
## time0  0.12660789 0.098274623
## time1  0.02191499 0.130944480
## time2  0.01515850 0.041841894
```

```
## time4  0.13955356 -0.054688208
## time5  0.18041288 -0.262282020
## time6  0.20031638 -0.331657958
## time7  0.19175406 -0.378492872
## time8  0.19495465 -0.366681179
## time9  0.21110968 -0.312703062
## time10 0.22705317 -0.229686062
## time11 0.23883449 -0.128725943
## time12 0.24462286 -0.061299611
## time13 0.24722050 -0.002188491
## time14 0.24585774  0.068679735
## time15 0.24219665  0.128475897
## time16 0.23803459  0.170456008
## time17 0.23570565  0.188679577
## time18 0.23537150  0.198603828
## time19 0.23539826  0.199006511
## time20 0.22860113  0.219922803
## time21 0.22918696  0.229053930
## time22 0.22507059  0.211992902
## time23 0.21127065  0.174308938
```

```r
### PC1 : 전반적인 승차객수, PC2 : 오전시간 대비 오후시간대의 승차객수(특히 오후 7시 이후 유흥지역이 활발히 운영

plot(pca_weekend_up$sdev^2, xlab = "Component number",
     ylab = "Component variance", type = "l", main = "Scree diagram")
```



```r
tmp2 <- dat_weekend_mat_up[,-1]
biplot(prcomp(tmp2, scale = TRUE), col = c("black", "darkgray"), xlim =
c(-0.5, 0.7), cex = 0.7, main = "Weekend Biplot(up)")
```



Group Selection(PCA, MDS 결과 참고)

- Group 1 : 평일 오전 하차시간대 비슷한 패턴을 가지는 지하철역들(근무지역 예상) 평일 오후 승차시간대 비슷한 패턴을 가지는 지하철역들
- Group 2 : 주말 오후 하차시간대 비슷한 패턴을 가지는 지하철역들(유흥지역 예상) 주말 늦은 오후 승차시간대 비슷한 패턴을 가지는 지하철역들
- Group 3 : 평일/주말 늦은 하차시간대와 평일/주말 이른 승차시간대 등 비슷한 패턴을 가지는 지하철역들 (주거지 예상)

0401 subwaydata / handled from Linux

```r
load(file = "subway0401.RData")
```

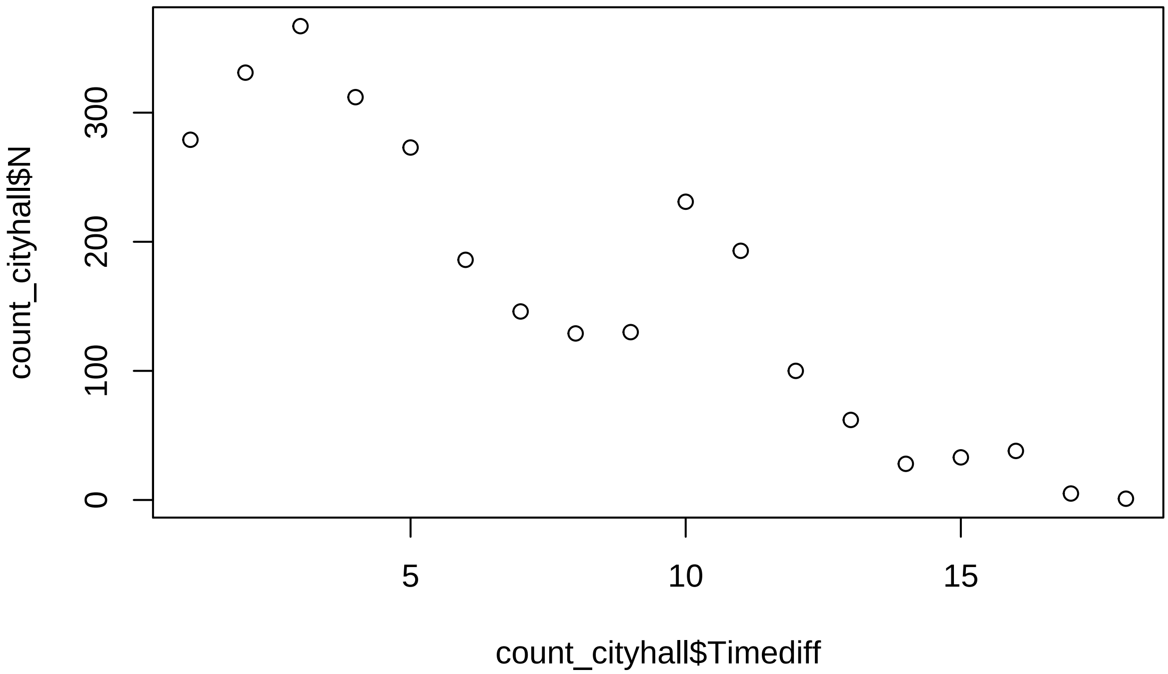
예시) 승,하차역이 시청역인 경우

```r
sub_cityhall_up <- subway0401[up_Name=="시청",]
sub_cityhall_up <- sub_cityhall_up%>%filter(Time>=15)
```

```r
sub_cityhall_down <- subway0401[down_Name=="시청",]
sub_cityhall_down <- sub_cityhall_down%>%filter(Time>=4)

sub_cityhall_down <- sub_cityhall_down%>%dplyr::select(ID, up_Name, down_Name, Time)
sub_cityhall_up <- sub_cityhall_up%>%dplyr::select(ID, Time, up_Name, down_Name)%>%rename(Time2=Time,do
sub_cityhall <- left_join(sub_cityhall_down, sub_cityhall_up)
```

```
## Joining, by = "ID"
```

```r
sub_cityhall <- na.omit(sub_cityhall)
sub_cityhall <- sub_cityhall%>%mutate(Timediff = Time2 - Time)
sub_cityhall_diff <- sub_cityhall %>% filter(Timediff>0)
count_cityhall <- sub_cityhall_diff%>%group_by(Timediff)%>%summarise(N=n())
plot(x = count_cityhall$Timediff, y = count_cityhall$N)
```



- 시청의 경우 18.04.01(일)이 주말에도 불구하고 bimodal한 형태를 보이는 것을 확인.

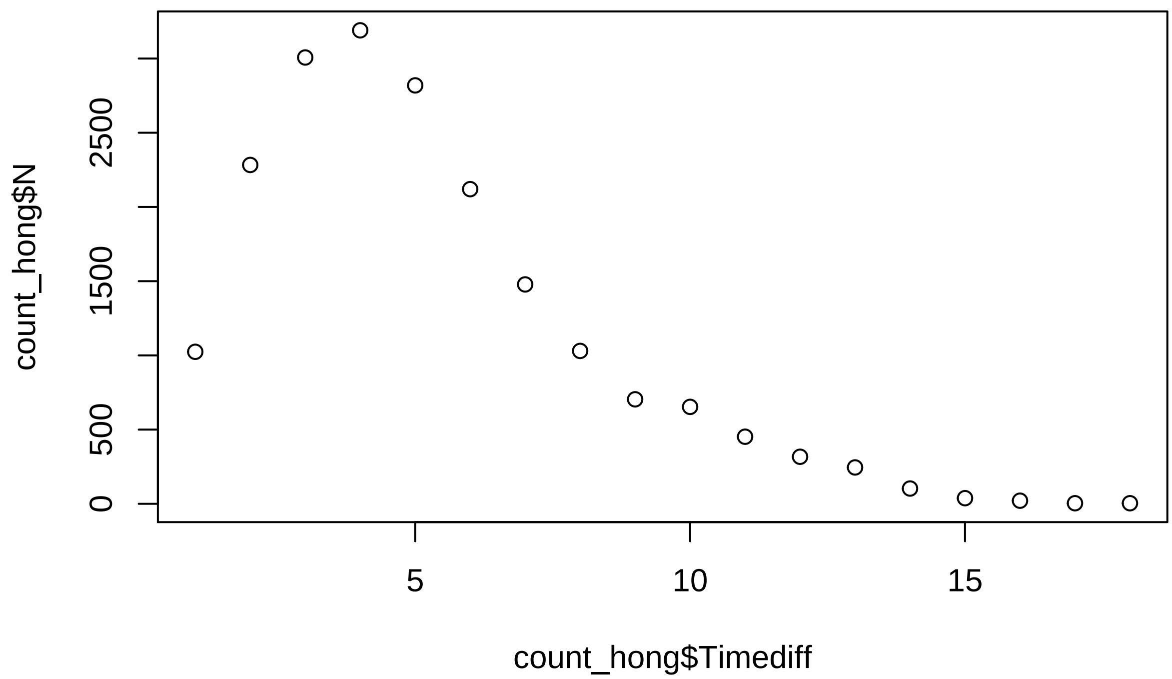
예시) 승,하차역이 홍대입구역인 경우

```r
sub_hong_up <- subway0401[up_Name=="홍대입구",]
sub_hong_up <- sub_hong_up%>%filter(Time>=15)
sub_hong_down <- subway0401[down_Name=="홍대입구",]
sub_hong_down <- sub_hong_down%>%filter(Time>=4)

sub_hong_down <- sub_hong_down%>%dplyr::select(ID, up_Name, down_Name, Time)
```

```
sub_hong_up <- sub_hong_up%>%dplyr::select(ID, Time, up_Name, down_Name)%>%rename(Time2=Time,down_Name2=
sub_hong <- left_join(sub_hong_down, sub_hong_up)
```

```
## Joining, by = "ID"
```

```
sub_hong <- na.omit(sub_hong)
sub_hong <- sub_hong%>%mutate(Timediff = Time2 - Time)
sub_hong_diff <- sub_hong %>% filter(Timediff>0)
count_hong <- sub_hong_diff%>%group_by(Timediff)%>%summarise(N=n())
plot(x = count_hong$Timediff, y = count_hong$N)
```



- 그림을 통하여 홍대입구역 정착시간의 분포가 카이제곱분포의 형태를 따르는 것을 확인

Future Study

1> 잔류시간 관련

- 직장인의 평균 근무시간을 지하철이용객에 한하여 추정
- 유흥밀집지역 이용객들의 평균 잔류시간 추정
- 시각화

2> 출/퇴근시간 관련

- subway_shortestpath를 통한 시간산출 결과를 이용하여 직장인의 평균 통근시간을 지하철이용객에 한하여 추정
- 유흥밀집지역 이용객들의 거주지에서 평균 이동시간

- 주요 밀집지역별 이용객의 거주지 - 밀집지역 거리 확인
- 시각화