Algorithm

Anandu R

5/31/2020

Importing necessary packages/Libraries

```
invisible(library(dplyr))
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
invisible(library(lubridate))
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
invisible(library(caTools))
invisible(library(data.table))
```

```
## data.table 1.12.8 using 4 threads (see ?getDTthreads).  Latest news: r-datatable.com

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week, yday, year

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```r
invisible(library(rpart))
invisible(library(rpart.plot))
invisible(library(ggplot2))
invisible(library(reshape2))
```

```
##
## Attaching package: 'reshape2'

## The following objects are masked from 'package:data.table':
##
##     dcast, melt
```

Generating the dataset

Data description:

- speed: Real-time speed of the bus in commute .
- dist_prev: The distance between the current bus and the previous bus on the same route.
- dist_next: The distance between the current bus and the next bus on the same route.
- crowd_curr: The number of passengers currently in the bus.
- crowd_next: The number of passengers in the next bus on the same route.
- schd_time: The scheduled arrival time specified for bus at previous stop on their route.
- arr_time: The actual arrival time of the bus at the previous stop on their route.
- on_time: Whether the bus arrived on time or not at the previous stop on their route.
- time_delay: The difference between the actual arrival time and the scheduled arrival time.

```r
set.seed(1)
speed = round(rnorm(1000,50,15),2)
dist_prev = abs(round(rnorm(1000,2,1),2))
dist_next = abs(round(rnorm(1000,2,1),2))
crowd_curr = rpois(1000,25)
crowd_next = rpois(1000,25)
schd_time = sample(seq(strptime('01/01/2018',format = "%d/%m/%Y"),
                       strptime('01/01/2019',format = "%d/%m/%Y"),
                       by="hour"), 1000, replace = T)
arr_time = schd_time+(rnorm(1000,300,350)*-1)
on_time = ifelse(difftime(arr_time,schd_time)<=0,1,0)
time_delay = difftime(arr_time,schd_time)
data = data.frame(crowd_curr,crowd_next,
                  dist_prev, dist_next,speed,
                  schd_time,arr_time,on_time,time_delay)
head(select(data,crowd_curr,crowd_next,on_time),10)
```

```
##    crowd_curr crowd_next on_time
## 1          28         27       0
## 2          26         24       1
## 3          31         21       1
## 4          20         28       1
## 5          27         21       0
## 6          23         21       0
## 7          22         23       1
```

```
## 8          34          37        0
## 9          32          21        1
## 10         27          15        0
```

Generating an algorithm to label the datasets

Each record is considered as a bus and the label is the indication given to the bus driver whether to maintain speed, decrease speed, or to increase represented by 0,1,2 respectively

```
data = data %>% mutate(.,indicate = with(.,case_when(
  (dist_next<1.8 & dist_prev<1.8 & crowd_next<25 & crowd_curr<28) ~ 0,
  (dist_next<1.8 & dist_prev<1.8 & crowd_next<25 & crowd_curr>28) ~ 0,
  (dist_next<1.8 & dist_prev<1.8 & crowd_next>25 & crowd_curr<28) ~ 2,
  (dist_next<1.8 & dist_prev<1.8 & crowd_next>25 & crowd_curr>28) ~ 0,
  (dist_next<1.8 & dist_prev>1.8 & crowd_next<25 & crowd_curr<28) ~ 1,
  (dist_next<1.8 & dist_prev>1.8 & crowd_next<25 & crowd_curr>28) ~ 1,
  (dist_next<1.8 & dist_prev>1.8 & crowd_next>25 & crowd_curr<28) ~ 2,
  (dist_next<1.8 & dist_prev>1.8 & crowd_next>25 & crowd_curr>28) ~ 1,
  (dist_next>1.8 & dist_prev<1.8 & crowd_next<25 & crowd_curr<28) ~ 2,
  (dist_next>1.8 & dist_prev<1.8 & crowd_next<25 & crowd_curr>28) ~ 0,
  (dist_next>1.8 & dist_prev<1.8 & crowd_next>25 & crowd_curr<28) ~ 2,
  (dist_next>1.8 & dist_prev<1.8 & crowd_next>25 & crowd_curr>28) ~ 2,
  (dist_next>1.8 & dist_prev>1.8 & crowd_next<25 & crowd_curr<28) ~ 0,
  (dist_next>1.8 & dist_prev>1.8 & crowd_next<25 & crowd_curr>28) ~ 0,
  (dist_next>1.8 & dist_prev>1.8 & crowd_next>25 & crowd_curr<28) ~ 2,
  (dist_next>1.8 & dist_prev>1.8 & crowd_next>25 & crowd_curr>28) ~ 0,
)))

head(select(data,crowd_curr,on_time,indicate),10)
```

```
##    crowd_curr on_time indicate
## 1          28       0       NA
## 2          26       1        1
## 3          31       1        0
## 4          20       1        2
## 5          27       0        0
## 6          23       0        2
## 7          22       1        0
## 8          34       0        0
## 9          32       1        0
## 10         27       0        1
```

The table below indicates the indications that each of the bus instances receive

```
data$indicate = factor(data$indicate,
                       levels=c(0,1,2),
                       labels = c("Maintain Speed",
                                  "Slow Down",
```

```
                                   "Speed Up"))
table(data$indicate)
```

```
##
## Maintain Speed      Slow Down       Speed Up
##            314            124            412
```

Thus we obtain the following observations from above table:

- Number of buses instructed to "Maintain Speed" : 314
- Number of buses instructed to "Slow Down" : 123
- Number of buses instructed to "Speed Up" : 412

Modelling a decision tree algorithm to make future scheduling

Splitting the data into train and test

```
set.seed(1)
split = sample.split(data$indicate, SplitRatio = 0.75)
train = data[split,]
test = data[!split,]
```

Creating a penalty matrix to avoid miscalculation

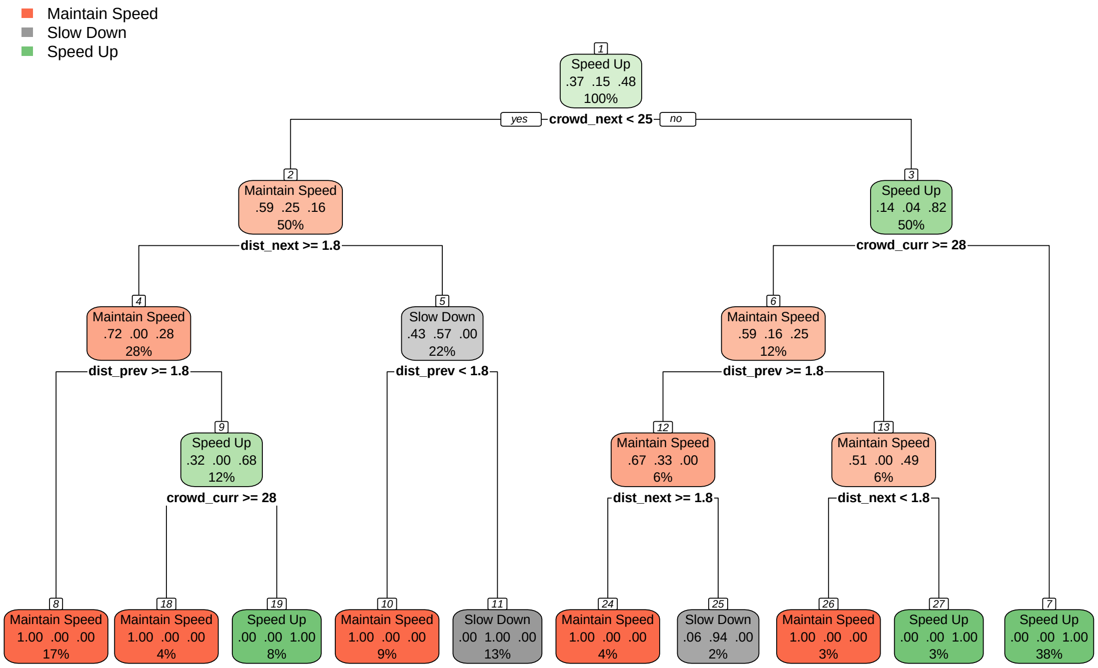
```
penalty.matrix <- matrix(c(1,1,0,10,0,10,0,0,0), byrow=TRUE, nrow=3)
```

Building the decision tree model with rpart

```r
dtree <- rpart(indicate~.,data=data,method = "class")
```

Visualizing the decision tree

```r
rpart.plot(dtree, nn=TRUE)
```



Using speed and on_time parameters

The speed and on_time parameters can be used for further analysis and using a regression model, we can provide the driver with recommended speed indication to maintain their schedule, and to keep them aware of whether they're on time or not

```r
head(select(data,speed,on_time))
```

```
##   speed on_time
## 1 40.60       0
## 2 52.75       1
## 3 37.47       1
## 4 73.93       1
## 5 54.94       0
## 6 37.69       0
```

Bus re-rerouting

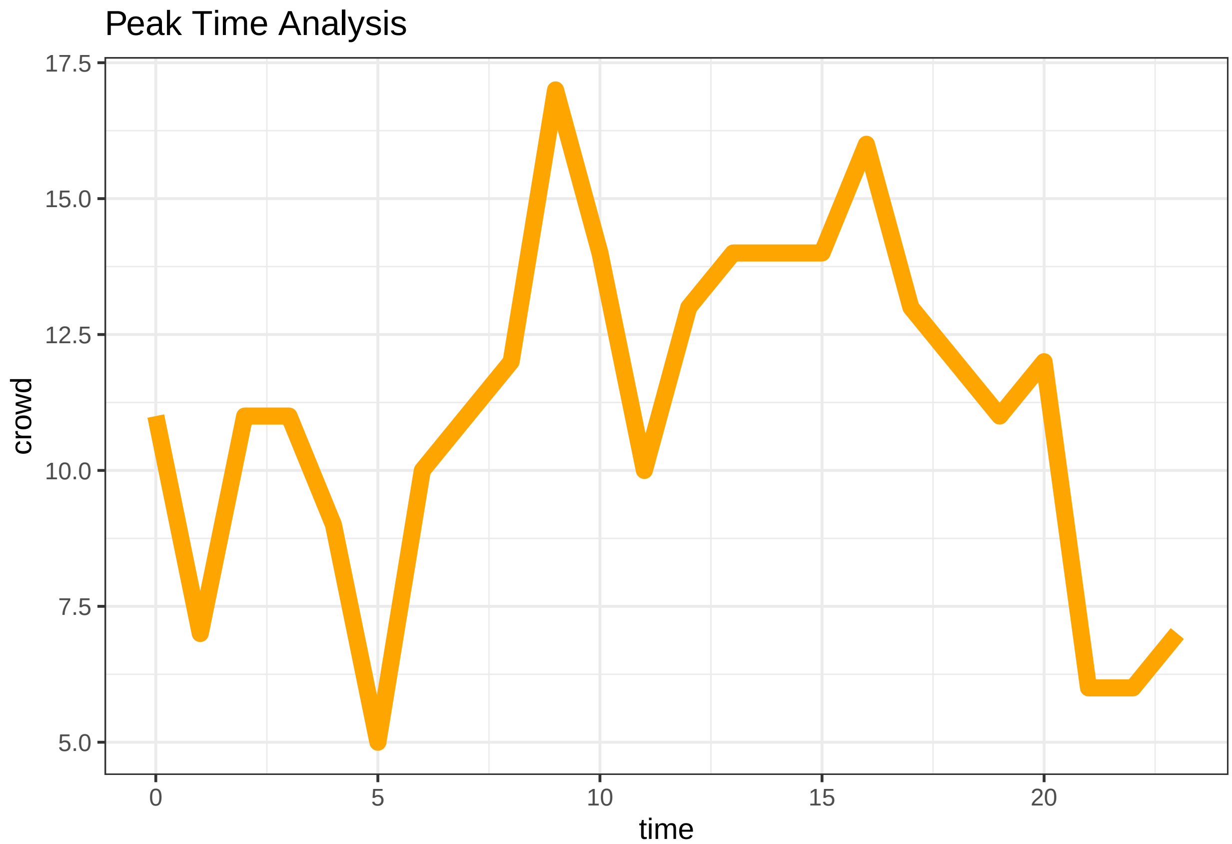
Consider each bus to be part of an area(a cluster), and each area has n number of buses, there are several routes in a given area, each route has predetermined number of buses plying through them.

Creating dummy dataset

```r
set.seed(0)
data_route_1 = data.frame(time = 0:23, crowd = c(round(rnorm(6,8,2)),round(rnorm(6,12,2)),round(rnorm(6
data_route_2 = data.frame(time = 0:23, crowd = c(round(rnorm(6,8,2)),round(rnorm(6,12,2)),round(rnorm(6
data_route_2[15,2] = 13
data_route_3 = data.frame(time = 0:23, crowd = c(round(rnorm(6,8,2)),round(rnorm(6,12,2)),round(rnorm(6
data_route_3[7,2] = 17
data_route_3[16,2] = 12
```

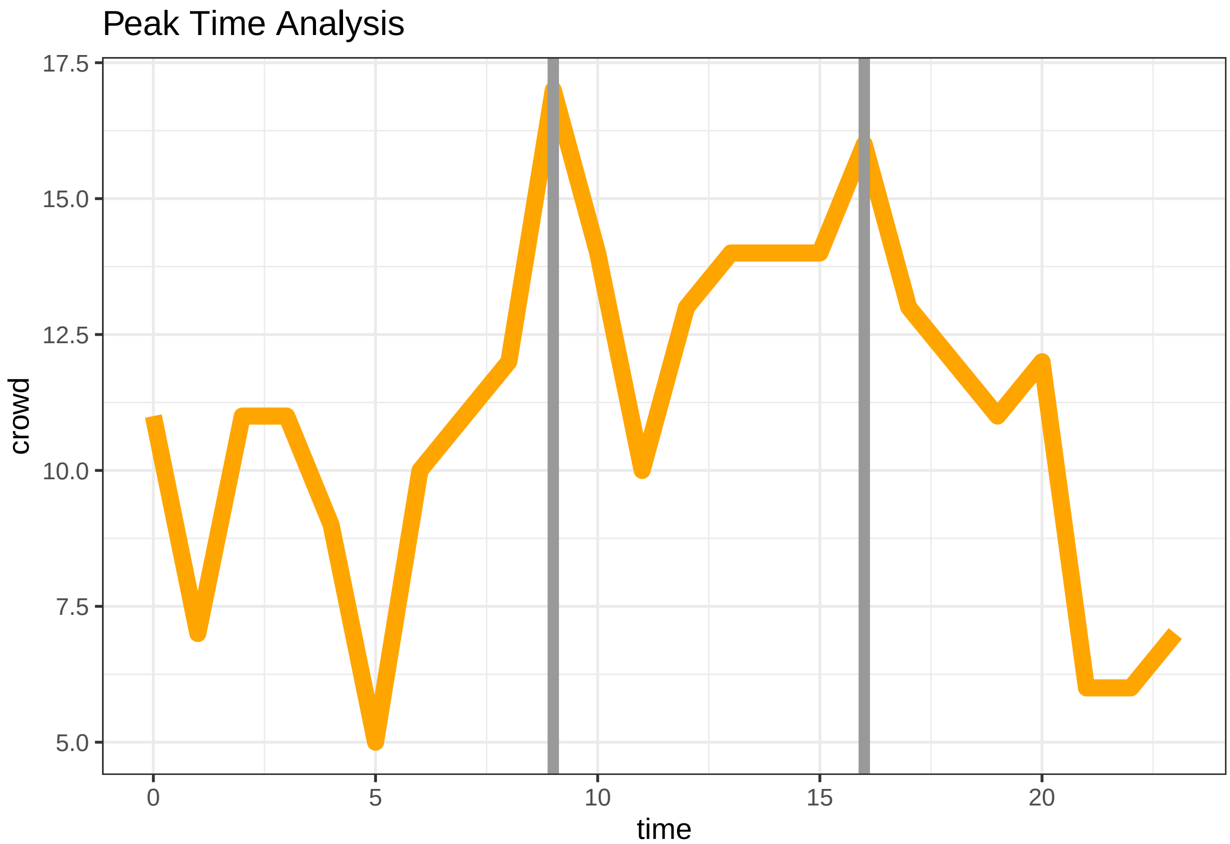
Using data analytics we determine the peak time of each route

```r
g1 = ggplot(
  data_route_1,
      aes(
        x = time,
        y = crowd
      )
)
g1 = g1 + geom_line(color = "orange", size = 3) + theme_bw() + ggtitle("Peak Time Analysis")
g1
```



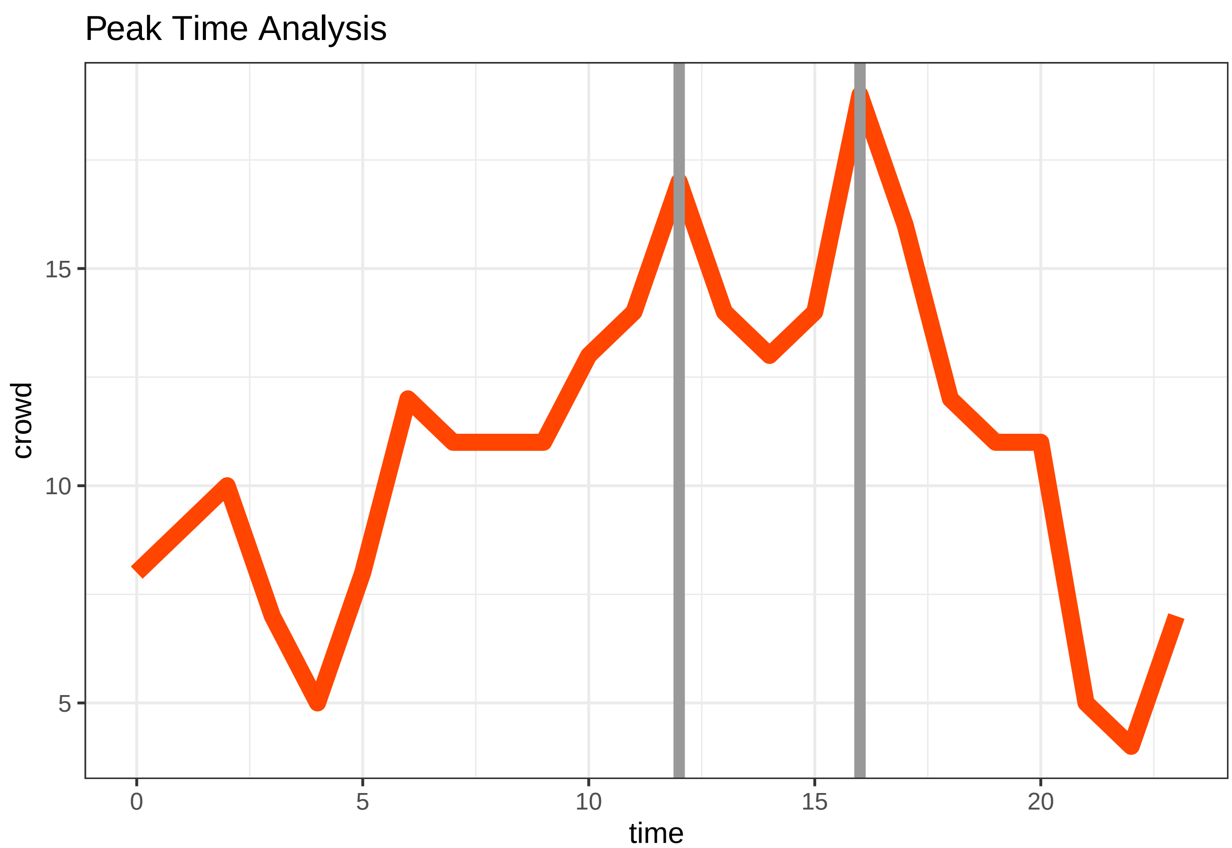
We can see that for route 1 the peak times are roughly around 9 am and 4 pm

```
g1 + geom_vline(xintercept =9,size = 2, col = "gray60") + geom_vline(xintercept =16, size = 2, col = "g
```

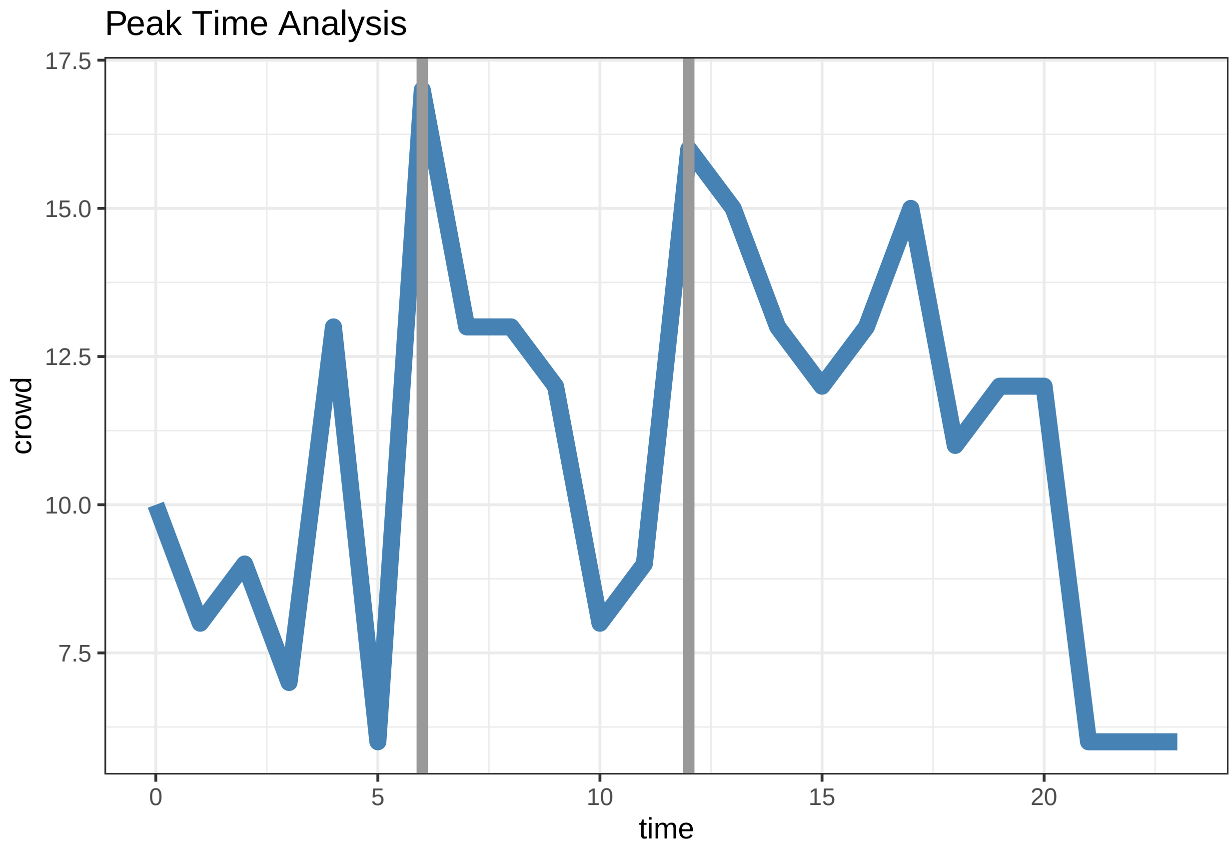


Similarly we have analytical data on other routes, here - route 2 and route 3, with their respective peak times.

```
g2 = ggplot(
  data_route_2,
      aes(
        x = time,
        y = crowd
      )
)
g2 = g2 + geom_line(color = "orangered", size = 3) + theme_bw() + ggtitle("Peak Time Analysis") + geom_v
g2
```



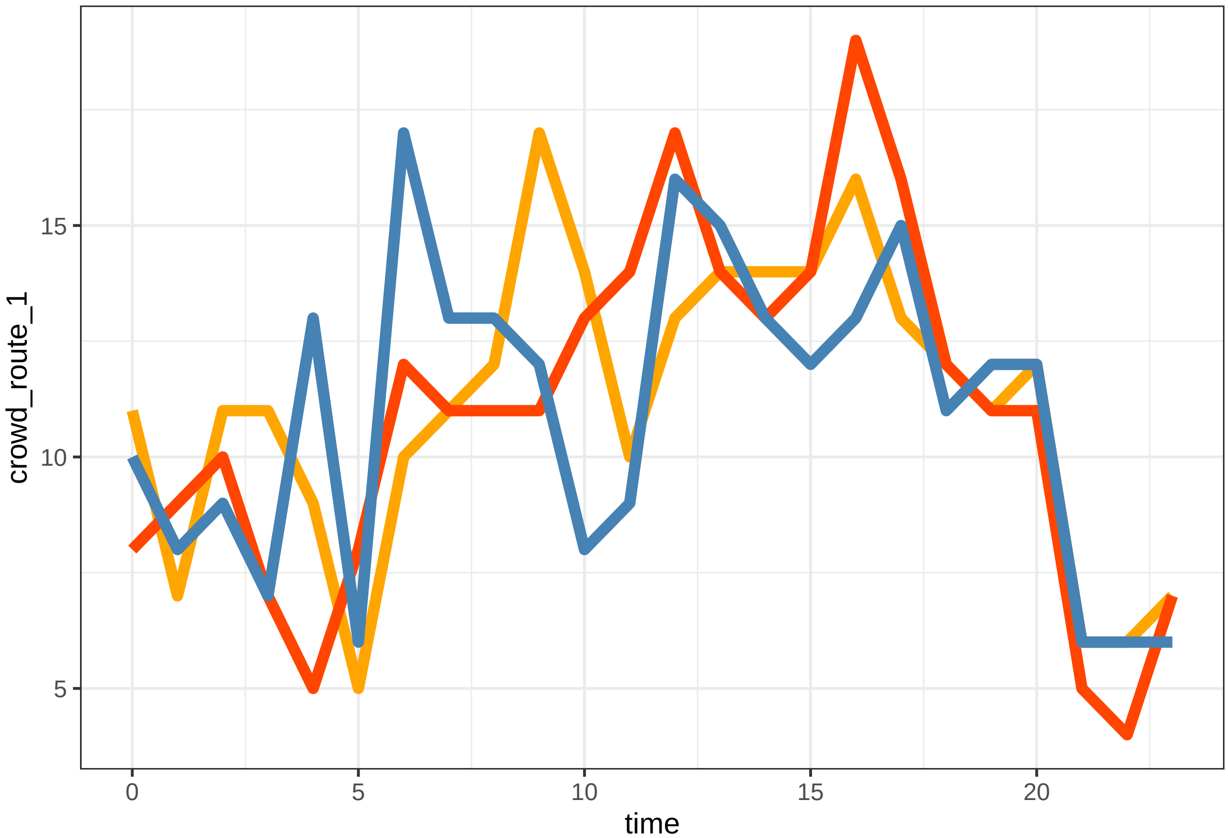

```
g2 = ggplot(
  data_route_3,
      aes(
        x = time,
        y = crowd
      )
)
g2 = g2 + geom_line(color = "steelblue", size = 3) + theme_bw() + ggtitle("Peak Time Analysis") + geom_
g2
```



We make a comparative study of the different peak times and schedules buses through various routes depending on the analytical data on those peak time.

```
data_all = data.frame(data_route_1$time,data_route_1$crowd,data_route_2$crowd,data_route_3$crowd)
names(data_all) = c("time","crowd_route_1","crowd_route_2","crowd_route_3")


g = ggplot(data_all, aes(x = time)) +
  geom_line(aes(y = crowd_route_1), color = "orange", size = 2) +
  geom_line(aes(y = crowd_route_2), color = "orangered", size = 2) +
  geom_line(aes(y = crowd_route_3), color = "steelblue", size = 2) +
  theme_bw()
g
```



crowd_route_1
15
10
5
0
5
10
15
20
time