Assignment4

Xiang Li

2019??9??24??

0. Preliminaries and overhead

```
rm(list=ls()); gc(); graphics.off()

##          used (Mb) gc trigger (Mb) max used (Mb)
## Ncells 515851 27.6    1163027 62.2   621665 33.3
## Vcells 986495  7.6    8388608 64.0  1600111 12.3

library(data.table)
library(ggplot2)
library(dplyr)
```

1. Problem 1

1.1

```
#create the function is_leap_year, this function will test whether a year is leap year or not.
Is_leap_Year<-function(year){
  if (!is.numeric(year) | year %% 1 != 0| year < 1950| year > 2050){
    print("Please type a four digit year between 1950 and 2050.")
  }
  else{
    if ((year %% 4 == 0 & year %% 100 != 0)|year %% 400 == 0){
      return(TRUE)
    }
    else{
      return(FALSE)
    }
  }
}
```

#1.2

```
#creathe the function second_Saturday, this function will print a list of the second Saturday for each month in a given year.
second_Saturday<-function(year){
  if (!is.numeric(year) | year %% 1 != 0) {
    print("Please type an integer.")
  }
  else{
    mon<-1:11
    for (m in mon) {
      start<-paste(as.character(year),"-",m,"-1",sep="")
      end<-paste(as.character(year),"-",m+1,"-1",sep="")
      wholemon<-seq.Date(from = as.Date(start,"%Y-%m-%d"),to =
as.Date(end,"%Y-%m-%d")-1, by = "day")
      sat<-vector()
      sat<-wholemon[weekdays(wholemon)=="Saturday"]
      print(paste("The second Saturday for month ",m," is ",sat[2],sep=""))
    }
    mon<-12
    start<-paste(as.character(year),"-",mon,"-1",sep="")
    end<-paste(as.character(year),"-",mon,"-31",sep="")
    wholemon<-seq.Date(from = as.Date(start,"%Y-%m-%d"),to = as.Date(end,"%Y-
%m-%d")-1, by = "day")
    sat<-vector()
    sat<-wholemon[weekdays(wholemon)=="Saturday"]
    print(paste("The second Saturday for month ",mon," is ",sat[2],sep=""))
  }
}
```

#1.3

```
#create the function day_Of_the_Week_in_N_days, the function will return the
weekday of the given give date after adding N days
day_Of_the_Week_in_N_days<-function(date,N){
  a<-as.Date(date,tryFormats = c("%Y-%m-%d", "%Y/%m/%d"))
  if (N %% 1 != 0) {
    print("Please type an integer.")
  }
  else{
    a<-a + N
    return(weekdays(a))
  }
}
```

2. Problem 2

#2.1

```
#this chunk load the flights dataset with two methods. The read.csv takes
about 30s while the data.table takes only 7s.
file_flights  = "C:/Users/xvidalberastain/Google
Drive/xavi_teaching/bus_111a_business_analytics_fall_2019/flight-
delays/flights.csv"
csv_flights   = read.csv(file_flights)    #30s
dta_flights   = fread(file = file_flights)#7s
```

#2.2

```
#this chunk order the two flights dataset with different methods. The basic
order method takes about 60s while the serorderv in data.table takes only 1s.
csv_flights =
csv_flights[order(csv_flights[,5],csv_flights[,6],csv_flights[,1],csv_flights
[,2],csv_flights[,3]),]#60s
col         = names(dta_flights)[c(5,6,1,2,3)]
dta_flights = setorderv(dta_flights,col,c(1,1,1,1,1),na.last = TRUE)#1s
```

#2.3

```
#create the column hour of the day
csv_flights$HOUR_OF_THE_DAY<-dta_flights$SCHEDULED_DEPARTURE%/%100
dta_flights$HOUR_OF_THE_DAY<-dta_flights$SCHEDULED_DEPARTURE%/%100
```

#2.3.a

```
#the chunk does the conditional calculating. The base aggregate function
takes 15 second, while data.table takes only 1 second.
head(aggregate(FLIGHT_NUMBER~HOUR_OF_THE_DAY+AIRLINE,csv_flights,FUN=length))
#15s

##   HOUR_OF_THE_DAY AIRLINE FLIGHT_NUMBER
## 1               0      AA          4395
## 2               1      AA          1240
## 3               2      AA            25
## 4               5      AA         17849
## 5               6      AA         44107
## 6               7      AA         63653

head(dta_flights[, .(number_of_flights=.N), by = .(AIRLINE,HOUR_OF_THE_DAY)])
#1s

##    AIRLINE HOUR_OF_THE_DAY number_of_flights
## 1:      AA               9             42148
## 2:      AA              12             45061
## 3:      AA              10             44484
## 4:      AA              23              5487
## 5:      AA              16             38466
## 6:      AA              14             42429
```

#2.3.b

```
#the chunk does the conditional calculating. The base aggregate function
takes 7 second, while data.table takes only 1 second.
head(aggregate(FLIGHT_NUMBER~HOUR_OF_THE_DAY+AIRLINE,csv_flights[which(csv_fl
ights$DEPARTURE_DELAY>15),],FUN=length)) #7s

##   HOUR_OF_THE_DAY AIRLINE FLIGHT_NUMBER
## 1               0      AA           598
## 2               1      AA           165
## 3               2      AA             9
## 4               5      AA          1122
## 5                  6       AA           2884
## 6                  7       AA           5574
```

```
head(dta_flights[DEPARTURE_DELAY>15][, .(number_of_flights=.N), by =
.(AIRLINE,HOUR_OF_THE_DAY)][order(AIRLINE,HOUR_OF_THE_DAY)]) #1s
```

```
##    AIRLINE HOUR_OF_THE_DAY number_of_flights
## 1:      AA               0               598
## 2:      AA               1               165
## 3:      AA               2                 9
## 4:      AA               5              1122
## 5:      AA               6              2884
## 6:      AA               7              5574
```

#2.3.c

```
#the chunk does the conditional calculating. The base aggregate function
takes 7 second, while data.table takes only 1 second.
head(aggregate(FLIGHT_NUMBER~HOUR_OF_THE_DAY+AIRLINE,csv_flights[which(csv_fl
ights$DEPARTURE_DELAY<15&csv_flights$DEPARTURE_DELAY>0),],FUN=length)) #7s
```

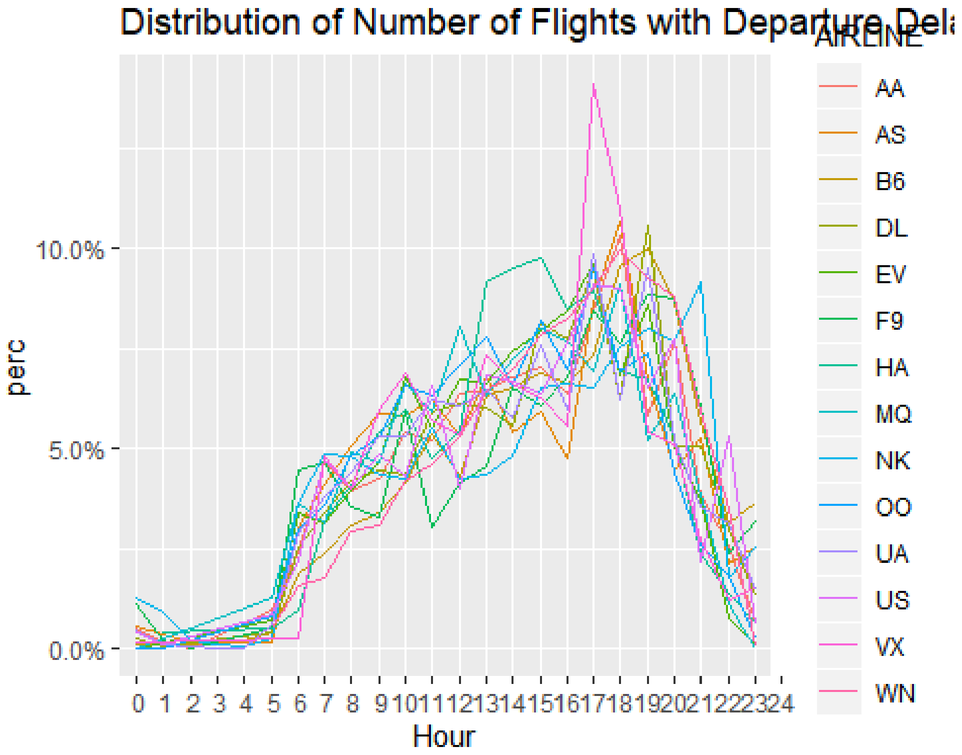
```
##   HOUR_OF_THE_DAY AIRLINE FLIGHT_NUMBER
## 1               0      AA           887
## 2               1      AA           230
## 3               2      AA             6
## 4               5      AA          2056
## 5               6      AA          4682
## 6               7      AA          7802
```

```
head(dta_flights[DEPARTURE_DELAY<15&DEPARTURE_DELAY>0][,
.(number_of_flights=.N), by =
.(AIRLINE,HOUR_OF_THE_DAY)][order(AIRLINE,HOUR_OF_THE_DAY)]) #1s
```

```
##    AIRLINE HOUR_OF_THE_DAY number_of_flights
## 1:      AA               0               887
## 2:      AA               1               230
## 3:      AA               2                 6
## 4:      AA               5              2056
## 5:      AA               6              4682
## 6:      AA               7              7802
```

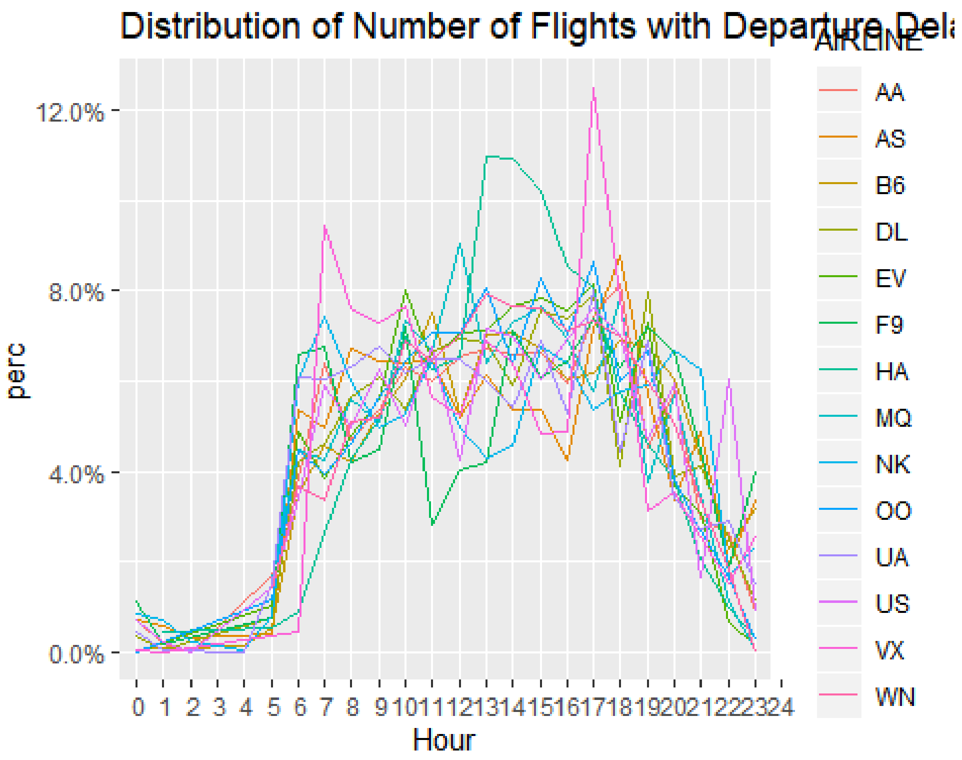
#2.3.d

```
#the chunk will visualize the distribution of number of flights with
departure delay less than 15 min or arrival delay greater than 15 min for
each airline.
#this part visualize the departure delay > 15min
ggplot(dta_flights %>%
         filter(DEPARTURE_DELAY>15&!is.na(DEPARTURE_DELAY))%>%
         group_by(AIRLINE,HOUR_OF_THE_DAY) %>%
         summarise(number_of_flights=n())%>%
         mutate(perc = number_of_flights/sum(number_of_flights)) %>%
         ungroup(),
         aes(x=HOUR_OF_THE_DAY,y=perc,group=AIRLINE,color=AIRLINE))+
  geom_line()+
  ggtitle(paste("Distribution of Number of Flights with Departure Delay
Greater Than 15min for Airlines"))+
  scale_y_continuous(labels = scales::percent)+
  scale_x_discrete(name ="Hour", limits=seq(0,24,1))
```



```
#this part visualize the 0min < departure < 15min delay
ggplot(dta_flights %>%

filter(DEPARTURE_DELAY>0&DEPARTURE_DELAY<15&!is.na(DEPARTURE_DELAY))%>%
         group_by(AIRLINE,HOUR_OF_THE_DAY) %>%
         summarise(number_of_flights=n())%>%
         mutate(perc = number_of_flights/sum(number_of_flights)) %>%
         ungroup(),
         aes(x=HOUR_OF_THE_DAY,y=perc,group=AIRLINE,color=AIRLINE))+
  geom_line()+
  ggtitle(paste("Distribution of Number of Flights with Departure Delay Less
Than 15min for Airlines"))+
  scale_y_continuous(labels = scales::percent)+
  scale_x_discrete(name ="Hour", limits=seq(0,24,1))
```



```
#Conclusion: In general, the percentage of delayed flights conditional on
hour of the day - most flights delay between 6 and 22, but conditional much
less on airlines. The distribution of delayed flights for different airlines
basically share similar mean and standard deviation with one or two outliers
at most.
```

2.4.a

```
#the chunk does the conditional calculating. The base aggregate function
takes 2.5 second, while data.table takes only 1 second.
tb1<-aggregate(FLIGHT_NUMBER~AIRLINE,csv_flights,FUN=length)
tb1$market_share<-tb1$FLIGHT_NUMBER/sum(tb1$FLIGHT_NUMBER)
head(tb1[,c(1,3)])#2.5s

##   AIRLINE market_share
## 1      AA   0.12475926
## 2      AS   0.02964748
## 3      B6   0.04589180
## 4      DL   0.15051884
## 5      EV   0.09829339
## 6      F9   0.01561003

head(unique(dta_flights[,
.(ttl=.N,AIRLINE)][,.(market_share=.N/ttl),by=AIRLINE]))#1s

##    AIRLINE market_share
## 1:      AA   0.12475926
## 2:      AS   0.02964748
## 3:      B6   0.04589180
## 4:      DL   0.15051884
## 5:      EV   0.09829339
## 6:      F9   0.01561003
```

2.4.b

```
#the chunk does the conditional calculating. The base aggregate function
takes 6 second, while data.table takes only 2.3 second.
tb2<-aggregate(FLIGHT_NUMBER~MONTH+AIRLINE,csv_flights,FUN=length)
tb3<-aggregate(FLIGHT_NUMBER~MONTH,csv_flights,FUN=length)
tb2<-merge(tb2,tb3,by.x = c(1),by.y = c(1))
tb2$market_share<-tb2$FLIGHT_NUMBER.x/tb2$FLIGHT_NUMBER.y
head(tb2[,c(1,2,5)])#6s
```

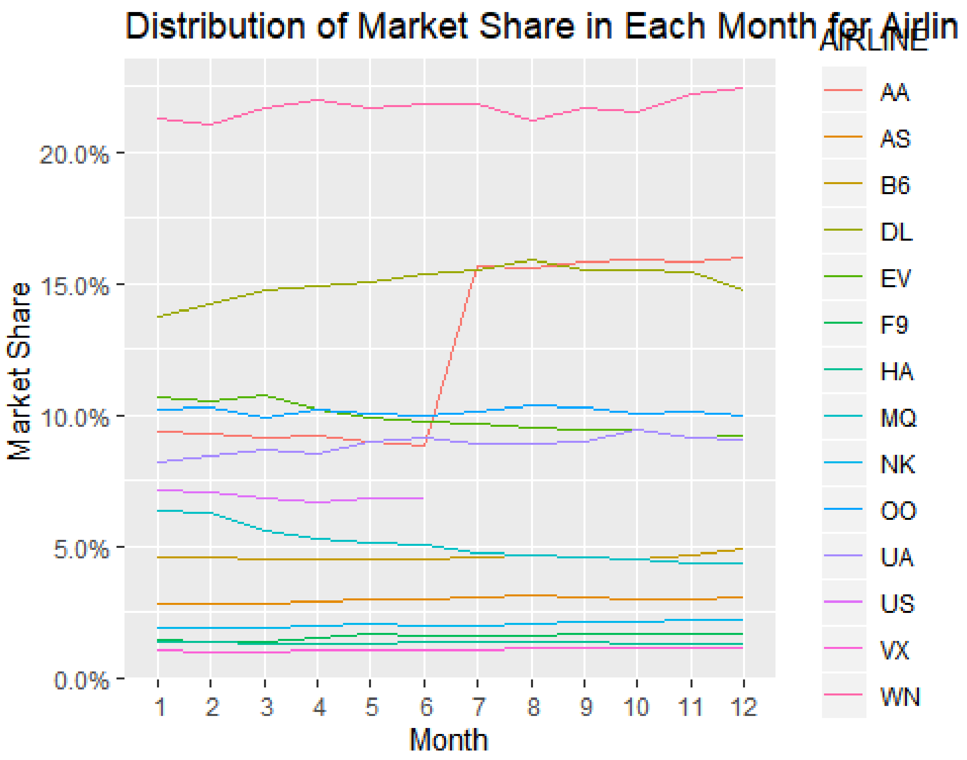
```
##   MONTH AIRLINE market_share
## 1     1      AA   0.09374894
## 2     1      UA   0.08169705
## 3     1      OO   0.10237718
## 4     1      NK   0.01860339
## 5     1      MQ   0.06362135
## 6     1      VX   0.01006664
```

```
dt1<-
merge(dta_flights[,.(ttl=.N),by=MONTH],dta_flights[,.(number_of_flights=.N),b
y=.(AIRLINE,MONTH)])
dt1<-dt1[,.(AIRLINE,MONTH,market_share=number_of_flights/ttl)]#2.3s
head(dt1)
```

```
##    AIRLINE MONTH market_share
## 1:      AA     1   0.09374894
## 2:      AS     1   0.02820830
## 3:      B6     1   0.04600952
## 4:      DL     1   0.13707529
## 5:      EV     1   0.10623064
## 6:      F9     1   0.01453078
```

2.4.b.cont****

```
#this chunk will visualize the distribution of number of flights in each
month for each airline
ggplot(dt1,aes(x=MONTH,y=market_share,group=AIRLINE,color=AIRLINE))+
  geom_line()+
  ggtitle(paste("Distribution of Market Share in Each Month for Airlines"))+
  scale_x_discrete(name ="Month", limits=seq(1,12,1))+
  scale_y_continuous(name="Market Share",labels = scales::percent)
```



```
#Conclusion: In general, the distribution of flights does not vary too much
by month. Some airlines own higher market share the whole year, while some
others own little. The invariance may because customers are likely to be
loyal to the firms they like, so they will not swith airlines in different
month. The gap among the market share for different airlines may be resulted
from their own policies, service levels or the discounts they offer to
customers. The difference between the market share for AA in the first half
of the year and in the second half of year may because it offer great
discounts in the second half of the year.
```