Predicting flight delay

Daniela

February 13, 2019

R Markdown

```
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
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
#### Step 1: Import Data.
setwd("C:/Users/Daniela Orovwiroro/Downloads/machine learning project/NEW FLIGHT")
# Import flights dataset after dowloading from the TSA website
flight = read.csv("flight.csv",sep=",", header=TRUE, stringsAsFactors = FALSE)
Airports = read.csv("airports.csv",sep=",", header=TRUE, stringsAsFactors = FALSE)
Airline = read.csv("airlines.csv",sep=",", header=TRUE, stringsAsFactors = FALSE)

# Examine the imported flight data.
dim(flight)
```

```
## [1] 20000    32
```

```r
# Review the first 6 rows of flight data.
head(flight)
```

```
##         X YEAR MONTH DAY DAY_OF_WEEK AIRLINE FLIGHT_NUMBER TAIL_NUMBER
## 1 4516166 2015    10   9           5      MQ          3176      N833MQ
## 2 4818743 2015    10  28           3      AA          1573      N489AA
## 3 4802060 2015    10  27           2      WN           432      N767SW
## 4 4415228 2015    10   2           5      WN           320      N387SW
## 5 4608226 2015    10  15           4      OO          2878      N868CA
## 6 4661210 2015    10  18           7      EV          2741      N671AE
##   ORIGIN_AIRPORT DESTINATION_AIRPORT SCHEDULED_DEPARTURE DEPARTURE_TIME
## 1            ABI                 DFW                 813            828
## 2            ABQ                 DFW                1208           1200
## 3            ABQ                 LAX                1100           1059
## 4            ABQ                 DAL                1755           1802
## 5            ABQ                 LAX                 605            647
## 6            AEX                 DFW                1344           1330
##   DEPARTURE_DELAY TAXI_OUT WHEELS_OFF SCHEDULED_TIME ELAPSED_TIME AIR_TIME
## 1              15        9        837             63           57       35
## 2              -8        8       1208            106           95       78
## 3              -1        6       1105            120          107       95
## 4               7        6       1808             95           88       77
## 5              42       31        718            120          145       99
## 6             -14       14       1344             82           67       47
##   DISTANCE WHEELS_ON TAXI_IN SCHEDULED_ARRIVAL ARRIVAL_TIME ARRIVAL_DELAY
## 1      158       912      13               916          925             9
## 2      569      1426       9              1454         1435           -19
## 3      677      1140       6              1200         1146           -14
## 4      580      2025       5              2030         2030             0
## 5      677       757      15               705          812            67
## 6      285      1431       6              1506         1437           -29
##   DIVERTED CANCELLED CANCELLATION_REASON AIR_SYSTEM_DELAY SECURITY_DELAY
## 1        0         0                                  NA             NA
## 2        0         0                                  NA             NA
## 3        0         0                                  NA             NA
## 4        0         0                                  NA             NA
## 5        0         0                                  25              0
## 6        0         0                                  NA             NA
##   AIRLINE_DELAY LATE_AIRCRAFT_DELAY WEATHER_DELAY
## 1            NA                  NA            NA
## 2            NA                  NA            NA
## 3            NA                  NA            NA
## 4            NA                  NA            NA
## 5            42                   0             0
## 6            NA                  NA            NA
```

Including Plots

You can also embed plots, for example:

```
##   MONTH DAY HDAYS
## 1    10   9     3
## 2    10  28    14
## 3    10  27    15
## 4    10   2    10
## 5    10  15     3
## 6    10  18     6
```

```
InputDays <- function(month,day){
    finalDays <- datesOfYear$HDAYS[datesOfYear$MONTH == month & datesOfYear$DAY == da
y] # Find which row to get
    return(finalDays)
}

flight$HDAYS = mapply(InputDays, flight$MONTH, flight$DAY)
head(flight)
```

```
##   MONTH DAY DAY_OF_WEEK AIRLINE ORIGIN_AIRPORT DESTINATION_AIRPORT
## 1    10   9           5      MQ            ABI                 DFW
## 2    10  28           3      AA            ABQ                 DFW
## 3    10  27           2      WN            ABQ                 LAX
## 4    10   2           5      WN            ABQ                 DAL
## 5    10  15           4      OO            ABQ                 LAX
## 6    10  18           7      EV            AEX                 DFW
##   DEPARTURE_TIME DEPARTURE_DELAY AIR_TIME DISTANCE ARRIVAL_TIME
## 1            828              15       35      158          925
## 2           1200              -8       78      569         1435
## 3           1059              -1       95      677         1146
## 4           1802               7       77      580         2030
## 5            647              42       99      677          812
## 6           1330             -14       47      285         1437
##   ARRIVAL_DELAY CANCELLED WEATHER_DELAY                 Airline_desc gain
## 1             9         0            NA American Eagle Airlines Inc.    6
## 2           -19         0            NA       American Airlines Inc.   11
## 3           -14         0            NA       Southwest Airlines Co.   13
## 4             0         0            NA       Southwest Airlines Co.    7
## 5            67         0             0        Skywest Airlines Inc.  -25
## 6           -29         0            NA  Atlantic Southeast Airlines   15
##   WEEKEND DEP_HOUR HDAYS
## 1       1        8     3
## 2       0       12    14
## 3       0       10    15
## 4       1       18    10
## 5       0        6     3
## 6       1       13     6
```

```
#Now, let us introduce one more column called DELAY_LABELED which has value 1
#if the arrival delay(ARR_DELAY) is more than 15 minutes and 0 if ARR_DELAY is less th
an 15 minutes.
#That means all flights which are arrived 15 minutes delayed are considered to be dela
yed.
flight$DELAY_LABELED = ifelse(flight$ARRIVAL_DELAY > 15, 1, 0)

#Next, edit the weather column assigning value 1 if the delay is more than 15 minutes
and 0 if delay is less than 15 minutes.
flight$WEATHER_DELAY[is.na(flight$WEATHER_DELAY)]=0
flight$WEATHER_DELAYs= ifelse(flight$WEATHER_DELAY>15,1,0)

flight$ARR_DEL15= ifelse(flight$ARRIVAL_DELAY>15,1,0)
flight$DEP_DEL15= ifelse(flight$DEPARTURE_DELAY>15,1,0)
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

```r
library(e1071)
#### Step 3: Prepare Training and Test Datasets.
## 80/20 split
# Filter records and create target variable 'gain'
model_data = flight %>% select( DELAY_LABELED,DAY_OF_WEEK, ARRIVAL_DELAY,WEEKEND,WEATH
ER_DELAY,DISTANCE, DEPARTURE_DELAY, Airline_desc,AIRLINE,gain)

tr = sample(nrow(model_data), round(nrow(model_data) * 0.8))
train = model_data[tr, ]
test = model_data[-tr, ]



###Step 4: Classification
library(mlbench)
library(partykit)
```

```
## Loading required package: grid
```

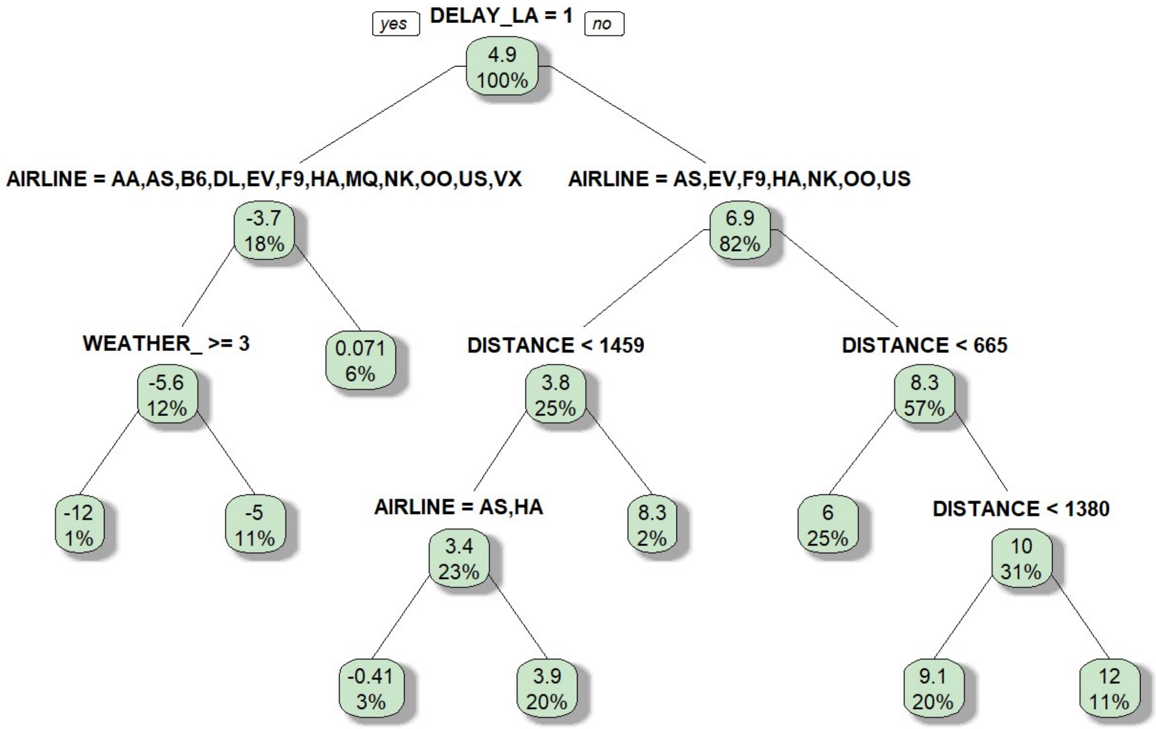
```
## Loading required package: libcoin
```

```
## Loading required package: mvtnorm
```

```r
library(rpart.plot)
```

```
## Loading required package: rpart
```

```r
library(RWeka)


##(a)using classification with Decision Tree.
# Build a decision tree model.
library(rpart) ## recursive partitioning
m =  rpart(gain ~ WEEKEND+DELAY_LABELED+AIRLINE +WEATHER_DELAY+DISTANCE, data = model_
data,
           cp=0)
pfit= prune(m, cp=m$cptable[9,"CP"])

prp(pfit,type=1,extra=100,fallen.leaves=F,shadow.col="darkgray",box.col=rgb(0.8,0.9,0.
8))
```




```
write.arff(model_data,"modeldata.arff")
```

```
##Step c: Classification using SVM model.
#Convert the outcome variable class to a factor:
train$DELAY_LABELED=factor(train$DELAY_LABELED)
#Build the model:
trctrl = trainControl(method = "repeatedcv", number = 10, repeats = 3)
set.seed(3233)

svm_Linear = train(DELAY_LABELED ~., data = train, method = "svmLinear",
                 trControl=trctrl,
                 preProcess = c("center", "scale"),
                 tuneLength = 10)
svm_Linear
```

```
## Support Vector Machines with Linear Kernel
##
## 15687 samples
##     9 predictor
##     2 classes: '0', '1'
##
## Pre-processing: centered (33), scaled (33)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 14118, 14118, 14118, 14118, 14118, 14119, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9985762  0.9952795
##
## Tuning parameter 'C' was held constant at a value of 1
```

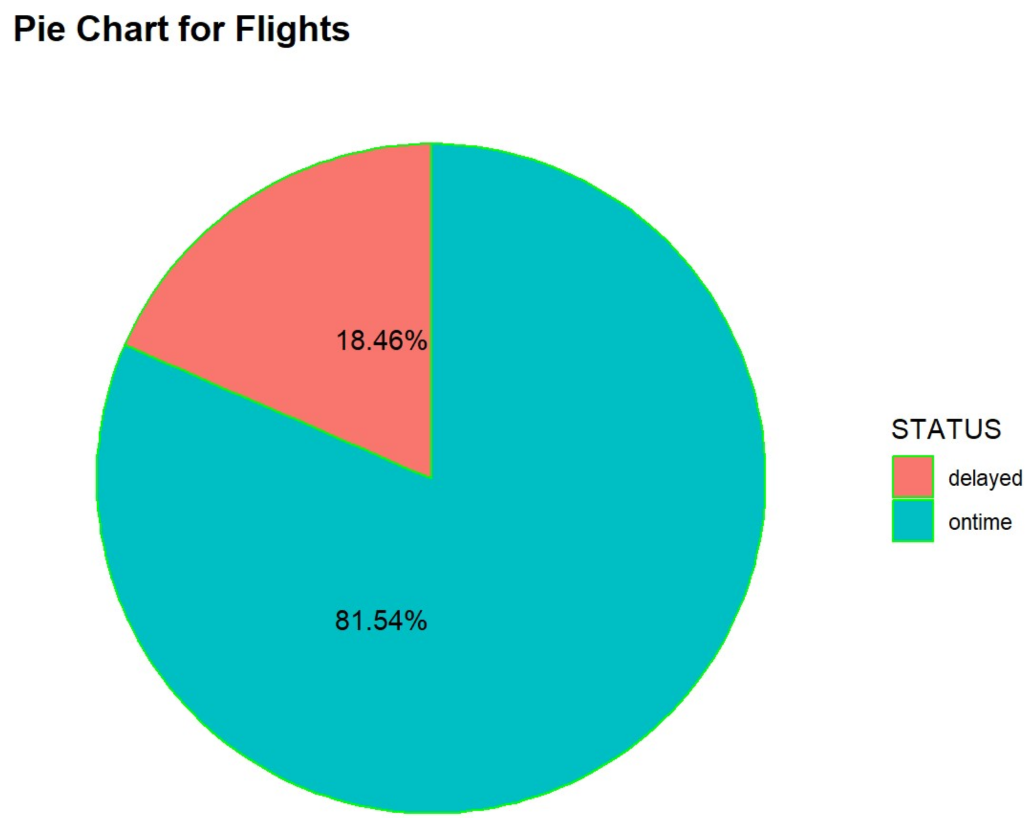
```
#predicting the result
test_pred = predict(svm_Linear, newdata = test)
```

```
ontime = flight[!is.na(flight$ARR_DEL15) & flight$ARR_DEL15!="" & !is.na(flight$DEP_DE
L15) & flight$DEP_DEL15!="",]
#Change the data class of the filtered data to enable data processing and running algo
rithms.
ontime$DESTINATION_AIRPORT =  as.factor(ontime$DESTINATION_AIRPORT)
ontime$ORIGIN_AIRPORT = as.factor(ontime$ORIGIN_AIRPORT)
ontime$DAY_OF_WEEK = as.factor(ontime$DAY_OF_WEEK)
ontime$DISTANCE = as.integer(ontime$DISTANCE)
ontime$CANCELLED = as.integer(ontime$CANCELLED)
ontime$DEP_HOUR = as.factor(ontime$DEP_HOUR)
ontime$AIRLINE= as.factor(ontime$AIRLINE)
ontime$Airline_desc= as.factor(ontime$Airline_desc)
ontime$ARR_DEL15 <- as.factor(ontime$ARR_DEL15)
ontime$DEP_DEL15 <-as.factor(ontime$DEP_DEL15)
ontime$gain <- as.factor(ontime$gain)
####Step6:Visualisation
# Summarize data by carrier

model_data$DELAY_LABELED=as.integer(model_data$DELAY_LABELED)
model_data$DAY_OF_WEEK=as.integer(model_data$DAY_OF_WEEK)
model_data$ARRIVAL_DELAY =  as.integer(model_data$ARRIVAL_DELAY)
model_data$WEEKEND=as.integer(model_data$WEEKEND)
model_data$WEATHER_DELAY=as.integer(model_data$WEATHER_DELAY)
model_data$DISTANCE=as.integer(model_data$DISTANCE)
model_data$DEPARTURE_DELAY=as.integer(model_data$DEPARTURE_DELAY)
model_data$Airline_desc=as.character(model_data$Airline_desc)
model_data$AIRLINE=as.character(model_data$AIRLINE)
model_data$gain= as.integer(model_data$gain)

new_flights= model_data %>%group_by(AIRLINE) %>%
  summarize(Airline_desc = min(Airline_desc), gain=mean(gain),
            DEPARTURE_DELAY=mean(DEPARTURE_DELAY)) %>%
  arrange(gain)

Flight.df=flight
#We create a new dataframe called delay which will have two columns, DELAY_LABELED an
d the count of it.
#Basically it will have a count of delayed flights and ontime flights.
#We will be using aggregate function of SparkR where we group the dataframe by DELAY_L
ABELED and calculating the count using n().
delay = Flight.df %>%
  group_by(DELAY_LABELED) %>%
  summarise(count=n())
#Introduce a new column called STATUS which will have value ontime if DELAY_LABELED i
s 0 and delayed if DELAY_LABELED is 1.
delay$STATUS = ifelse(delay$DELAY_LABELED == 0, "ontime", "delayed")

#Delete a first column DELAY_LABELED because we do not need it anymore.
```

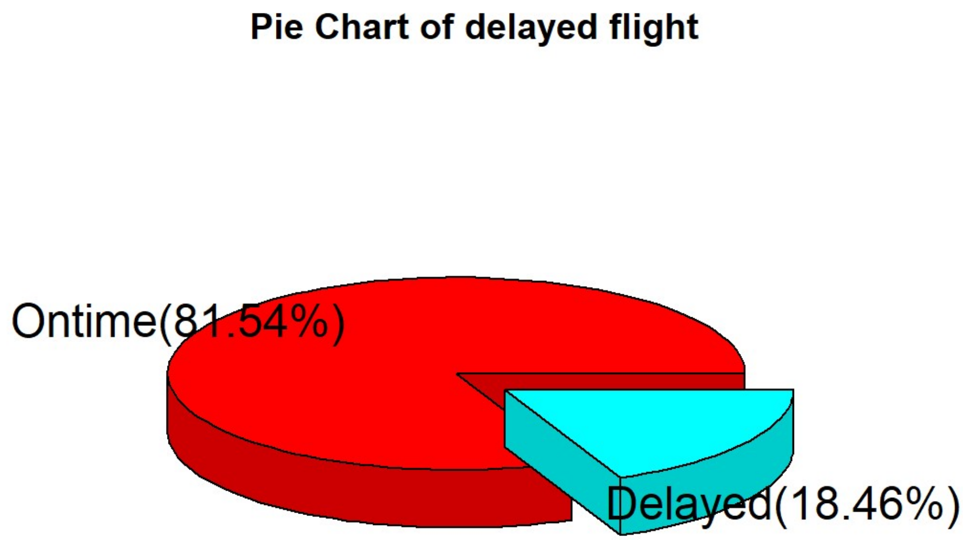
```r
delay = delay[,-1]

#Add Percentage as one more column to this new dataframe.
delay$Percentage = (delay$count / sum(delay$count)) * 100
delay$Percentage = round(delay$Percentage,2)
head(delay)
```

```
## # A tibble: 2 x 3
##   count STATUS  Percentage
##   <int> <chr>        <dbl>
## 1 15990 ontime        81.5
## 2  3619 delayed       18.5
```

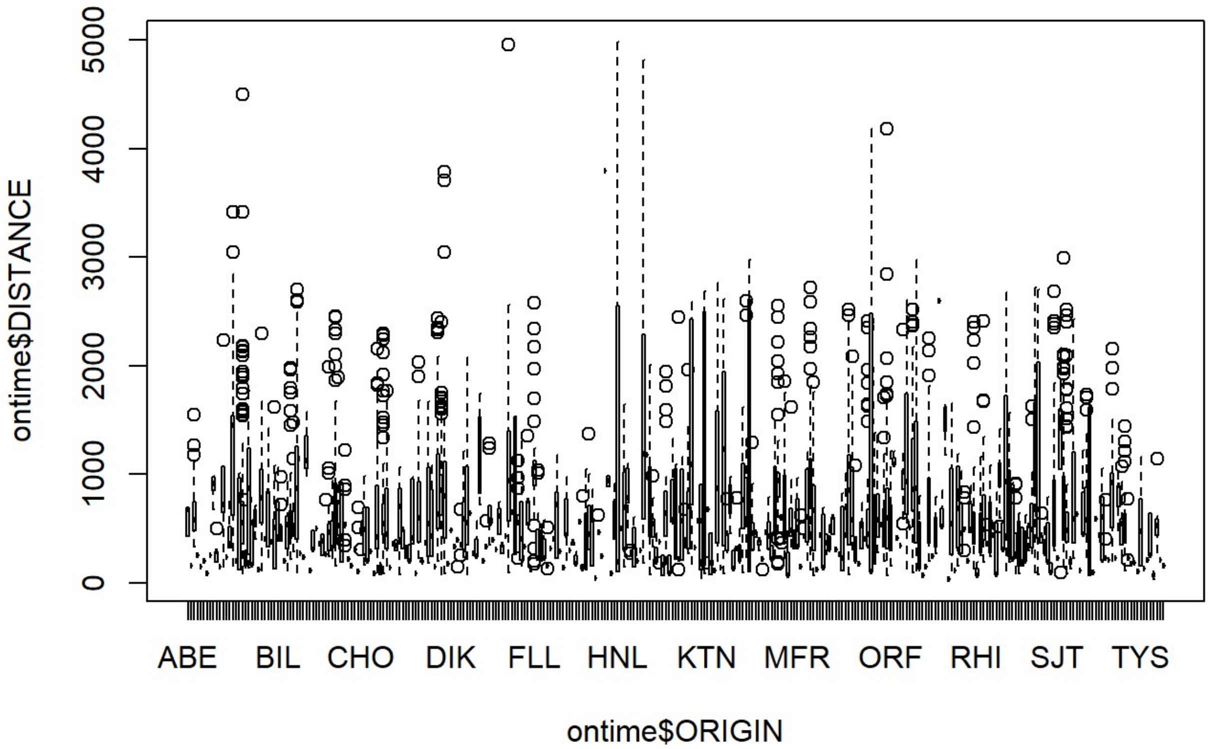
```r
blank_theme = theme_minimal()+
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),
    panel.border = element_blank(),
    panel.grid=element_blank(),
    axis.ticks = element_blank(),
    plot.title=element_text(size=14, face="bold")
  )
#We will draw a pie chart showing the percentage of delayed and ontime flights.
ggplot(delay, aes(x="",y=Percentage,fill=STATUS)) + geom_bar(stat="identity",width=1,c
olour="green") + coord_polar(theta="y",start=0) + blank_theme + ggtitle("Pie Chart fo
r Flights") + theme(axis.text.x=element_blank()) + geom_text(aes(y = Percentage/2,labe
l = paste0(Percentage,"%"),hjust=2))
```



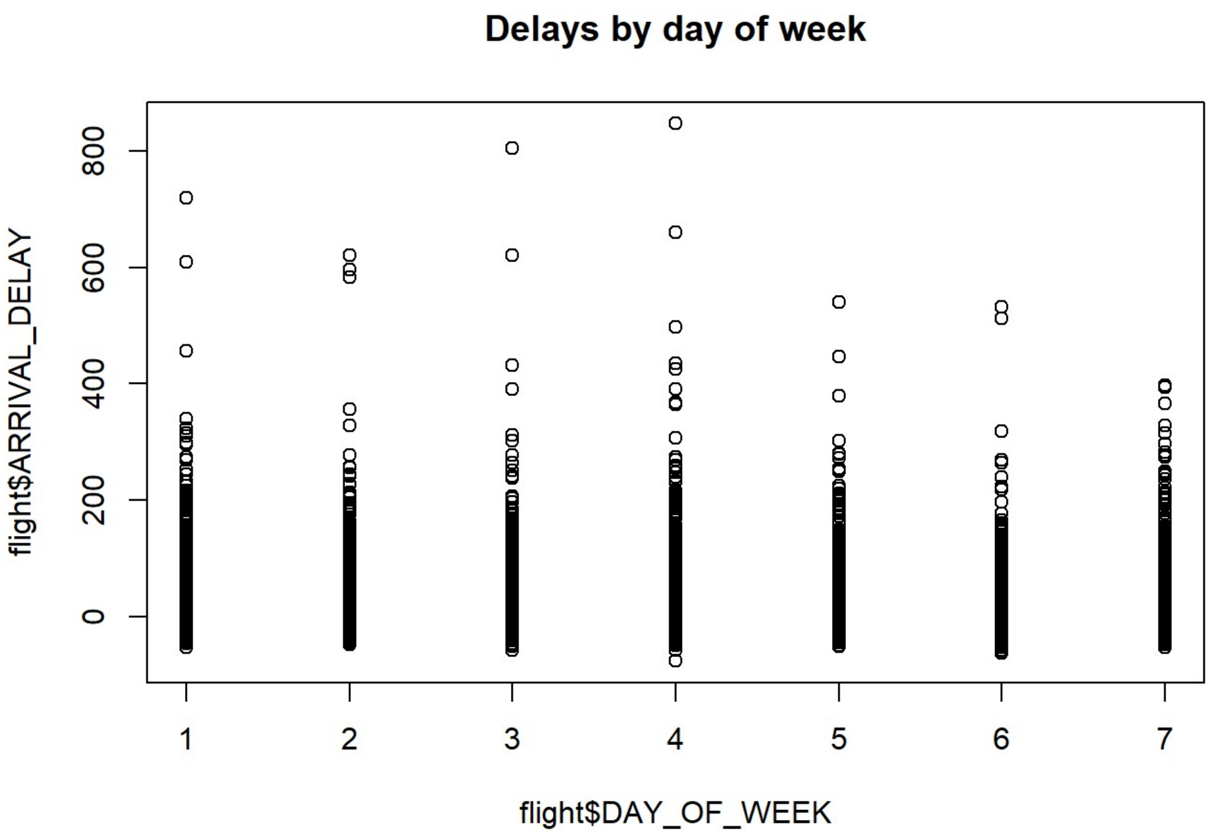

```r
library(plotrix)
slices <- delay$Percentage
lbls <- c("Ontime(81.54%)", "Delayed(18.46%)")

pie3D(slices,labels=lbls,explode=0.1,
    main="Pie Chart of delayed flight ")
```



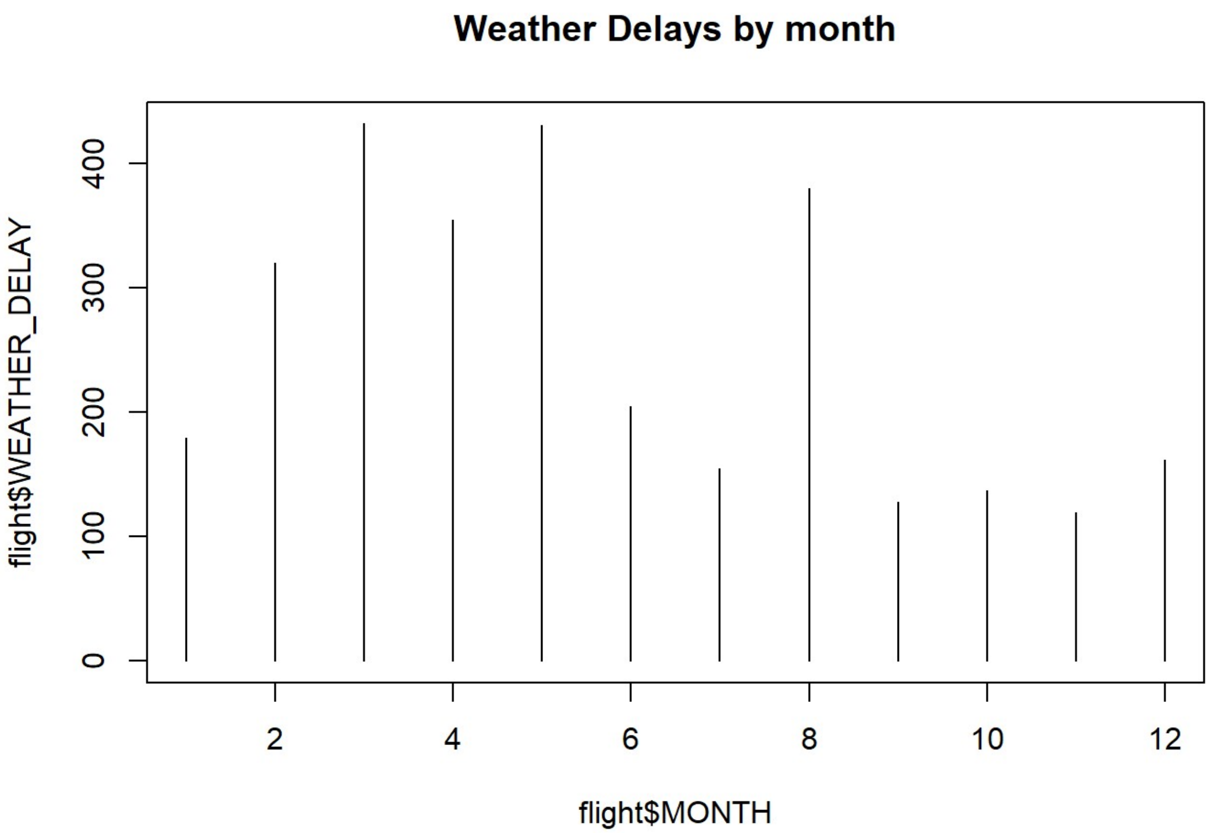

```
#The following plot shows when flights show how far destination cities are from a give
n originating airport. This is of importance as longer the flight, airlines can make u
p time in the air.
plot(ontime$DISTANCE ~ ontime$ORIGIN)
```




```
plot(flight$ARRIVAL_DELAY ~ flight$DAY_OF_WEEK,main= "Delays by day of week")
```




```
plot(flight$WEATHER_DELAY~flight$MONTH,type="h", main= "Weather Delays by month")
```


Weather Delays by month
flight$WEATHER_DELAY
0
100
200
300
400
2
4
6
8
10
12
flight$MONTH

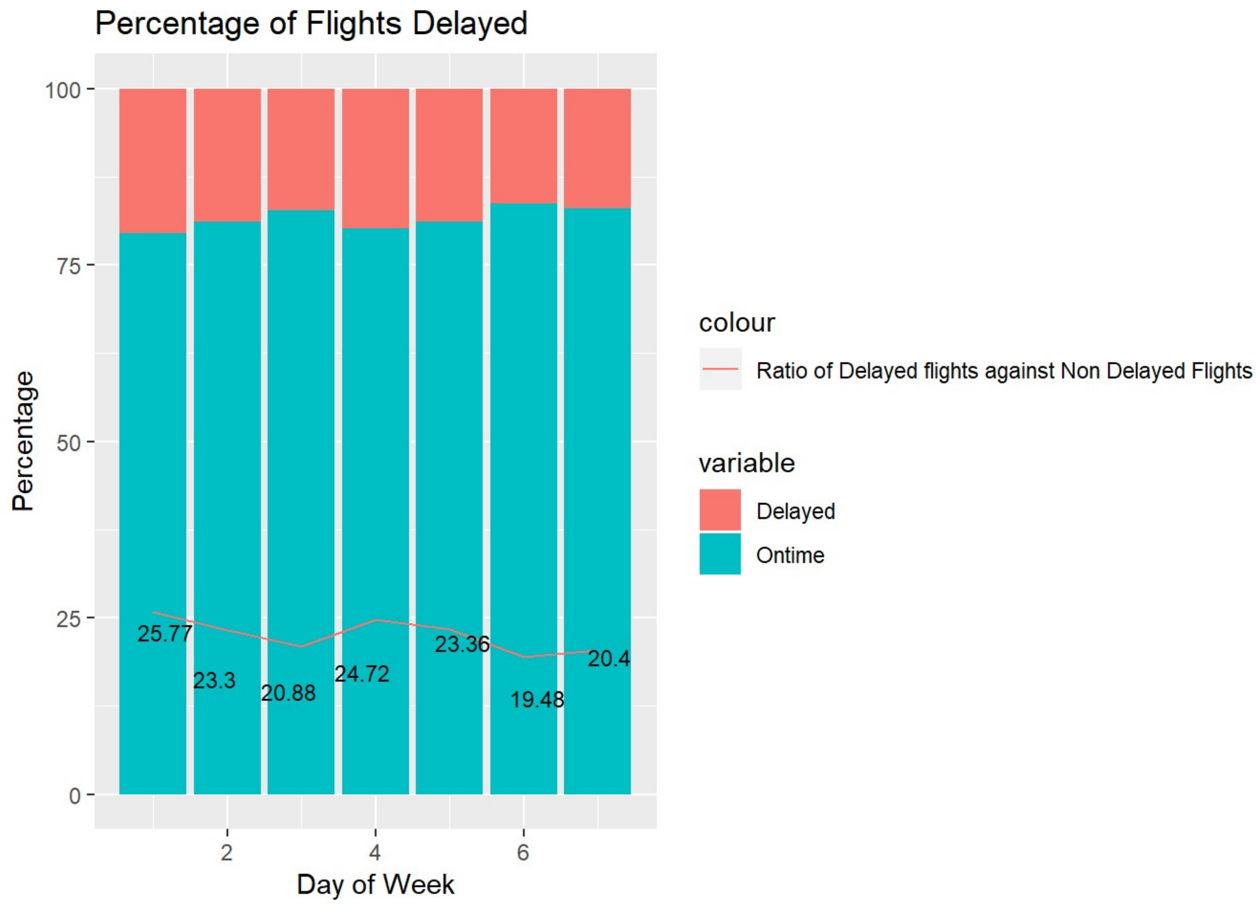
```
#Let us explore what effect Day_Of_Week has on the dataset. We will create two new dat
aframes called delay_flights and non_delay_flights which will have details for delaye
d and ontime flights respectively.
delay_flights = filter(Flight.df,Flight.df$DELAY_LABELED == 1)
non_delay_flights = filter(Flight.df,Flight.df$DELAY_LABELED == 0)
#Next, we will find the count of delayed and ontime flights grouped by Day_Of_Week.
delay_flights_count = delay_flights %>% group_by(DAY_OF_WEEK)%>%  summarise(count=n())
non_delay_flights_count = non_delay_flights %>% group_by(DAY_OF_WEEK)%>%  summarise(co
unt=n())

#Now, we can merge both delay_flights_count and non_delay_flights_count dataframes.
dayofweek_count = merge(x = delay_flights_count, y = non_delay_flights_count, by = "DA
Y_OF_WEEK", all.x = TRUE)
names(dayofweek_count)[names(dayofweek_count) == 'count.x'] = 'DELAY_COUNT'
names(dayofweek_count)[names(dayofweek_count) == 'count.y'] = 'NON_DELAY_COUNT'

#Introduce two columns, Delayed and Ontime, which have the percentage values for DELAY
_COUNT and NON_DELAY_COUNT respectively.
dayofweek_count$Delayed = (dayofweek_count$DELAY_COUNT/(dayofweek_count$DELAY_COUNT+da
yofweek_count$NON_DELAY_COUNT)) * 100

dayofweek_count$Ontime = (dayofweek_count$NON_DELAY_COUNT/(dayofweek_count$DELAY_COUNT
+dayofweek_count$NON_DELAY_COUNT)) * 100

dayofweek_count = dayofweek_count[,-2:-3]

#Next, add one more column which represents the ratio of delayed flights against ontim
e flights.
dayofweek_count$Ratio = dayofweek_count$Delayed/dayofweek_count$Ontime * 100

dayofweek_count$Ratio = round(dayofweek_count$Ratio,2)


#Now, if you look closely, our data is in wide format. The data is said to be in wide
format if there
#is one observation row per subject with each measurement present as a different varia
ble. We have to
#change it to long format which means there is one observation row per measurement thu
s multiple rows
#per subject. In R, we use reshape to do this:
library(reshape2)
DF1 = melt(dayofweek_count, id.var="DAY_OF_WEEK")
DF1$Ratio = DF1[15:21,3]
DF1
```

```
##    DAY_OF_WEEK variable    value Ratio
## 1            1  Delayed 20.49152 25.77
## 2            2  Delayed 18.89655 23.30
## 3            3  Delayed 17.27400 20.88
## 4            4  Delayed 19.81800 24.72
## 5            5  Delayed 18.93939 23.36
## 6            6  Delayed 16.30525 19.48
## 7            7  Delayed 16.94475 20.40
## 8            1   Ontime 79.50848 25.77
## 9            2   Ontime 81.10345 23.30
## 10           3   Ontime 82.72600 20.88
## 11           4   Ontime 80.18200 24.72
## 12           5   Ontime 81.06061 23.36
## 13           6   Ontime 83.69475 19.48
## 14           7   Ontime 83.05525 20.40
## 15           1    Ratio 25.77000 25.77
## 16           2    Ratio 23.30000 23.30
## 17           3    Ratio 20.88000 20.88
## 18           4    Ratio 24.72000 24.72
## 19           5    Ratio 23.36000 23.36
## 20           6    Ratio 19.48000 19.48
## 21           7    Ratio 20.40000 20.40
```

```
#We will change this dataframe just to make the plot more clearer.
DF1 = DF1[-15:-21,]
DF1[8:14,4] = NA

#Next, run the following line to see the stacked bar chart:
library(ggrepel)

ggplot(DF1, aes(x=DAY_OF_WEEK,y=value,fill=variable)) + geom_bar(stat="identity") + ge
om_path(aes(y=Ratio,color="Ratio of Delayed flights against Non Delayed Flights")) + g
eom_text_repel(aes(label=Ratio), size = 3) + ggtitle("Percentage of Flights Delayed")
+ labs(x="Day of Week",y="Percentage")
```

```
## Warning: Removed 7 rows containing missing values (geom_path).
```

```
## Warning: Removed 7 rows containing missing values (geom_text_repel).
```



Percentage of Flights Delayed
Percentage
100
75
50
25
0
25.77
23.3
20.88
24.72
23.36
19.48
20.4
2
4
6
Day of Week
colour
Ratio of Delayed flights against Non Delayed Flights
variable
Delayed
Ontime

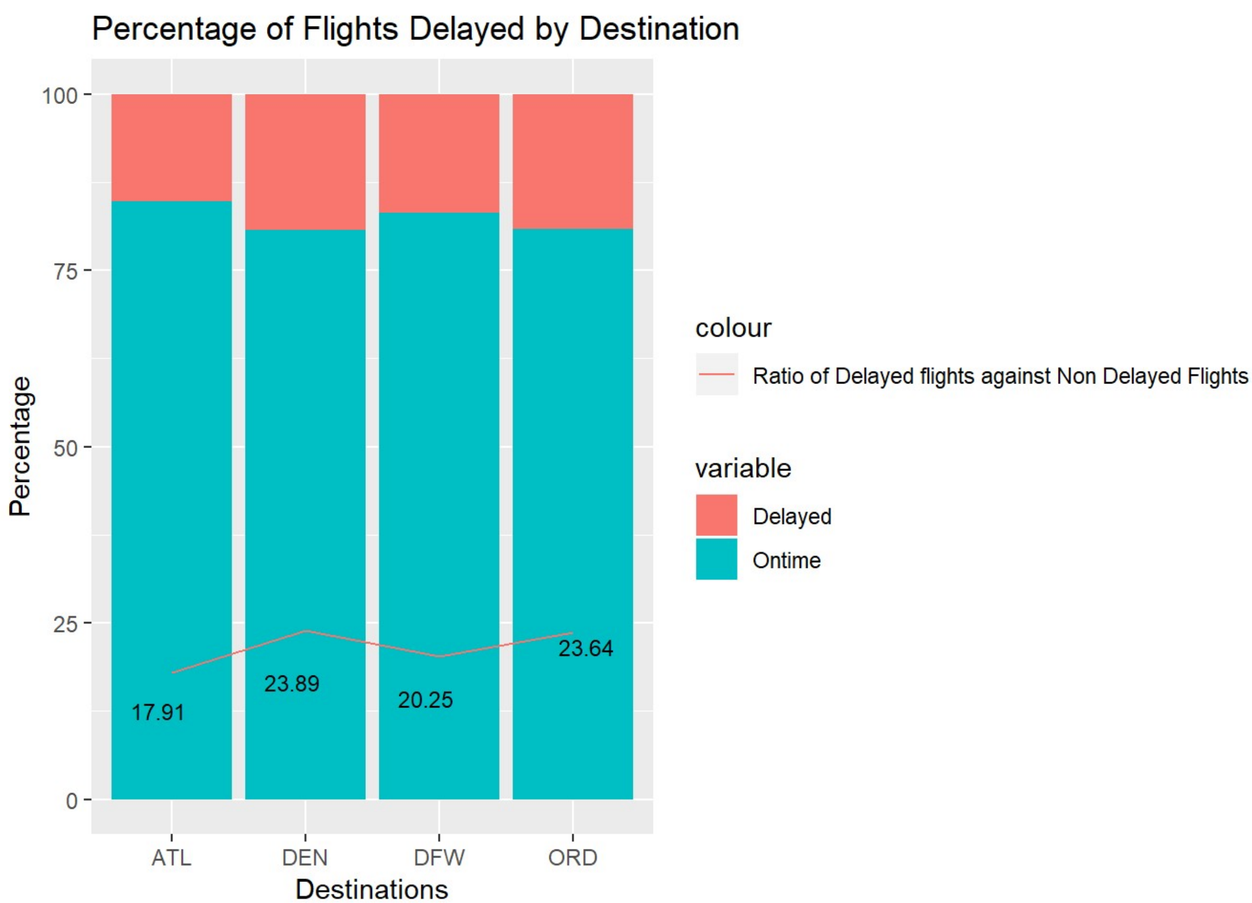
```
#As you can see here, most delays are happening on Tuesday and Saturday. It drops duri
ng the start of the weekend but again rises up by Sunday.
#Now we will look over Destination effect on the delays,
# Summarize data by carrier
new_dest_delay_flights= delay_flights %>% group_by(DESTINATION_AIRPORT) %>%summarise(c
ount=n())
new_dest_non_delay_flights= non_delay_flights %>% group_by(DESTINATION_AIRPORT) %>%sum
marise(count=n())

#Create two new dataframes from delay_flights and non_delay_flights dataframes respect
ively which will have the count of flights specific to some Destinations like LAX, SF
O, HNL, PDX.
destination_delay_count = delay_flights %>% group_by(DESTINATION_AIRPORT)%>%  summaris
e(count=n())
destination_delay_count = destination_delay_count[(destination_delay_count$DESTINATION
_AIRPORT == "ATL" | destination_delay_count$DESTINATION_AIRPORT == "ORD" | destination
_delay_count$DESTINATION_AIRPORT == "DFW" | destination_delay_count$DESTINATION_AIRPOR
T == "DEN") ,]

destination_non_delay_count = non_delay_flights %>% group_by(DESTINATION_AIRPORT)%>%
summarise(count=n())
destination_non_delay_count = destination_non_delay_count[(destination_non_delay_count
$DESTINATION_AIRPORT == "ATL" | destination_non_delay_count$DESTINATION_AIRPORT == "OR
D" | destination_non_delay_count$DESTINATION_AIRPORT == "DFW" | destination_non_delay_
count$DESTINATION_AIRPORT == "DEN") ,]

#Lets merge these two new dataframes into one.
destination_count = merge(x = destination_delay_count, y = destination_non_delay_coun
t, by = "DESTINATION_AIRPORT", all.x = TRUE)
names(destination_count)[names(destination_count) == 'count.x'] = 'DELAY_COUNT'
names(destination_count)[names(destination_count) == 'count.y'] = 'NON_DELAY_COUNT'

destination_count$Delayed = (destination_count$DELAY_COUNT/(destination_count$DELAY_CO
UNT+destination_count$NON_DELAY_COUNT)) * 100
destination_count$Ontime = (destination_count$NON_DELAY_COUNT/(destination_count$DELAY
_COUNT+destination_count$NON_DELAY_COUNT)) * 100
destination_count = destination_count[,-2:-3]

#Introduce one more column called Ratio which has the proportion of delayed flights ag
ainst ontime flights on the four aforementioned destinations
destination_count$Ratio = destination_count$Delayed/destination_count$Ontime * 100
destination_count$Ratio = round(destination_count$Ratio,2)

#As earlier, let us melt down this dataframe too to create a stacked bar chart. Use me
lt function of reshape package.
DF2 = melt(destination_count, id.var="DESTINATION_AIRPORT")
DF2$Ratio = DF2[9:12,3]
```

```
DF2 = DF2[-9:-12,]
DF2[5:8,4] = NA
```

```
#Draw a stacked bar chart:
ggplot(DF2, aes(x=DESTINATION_AIRPORT,y=value,fill=variable)) + geom_bar(stat="identit
y") + geom_path(aes(y=Ratio,color="Ratio of Delayed flights against Non Delayed Flight
s"),group = 1) + geom_text_repel(aes(label=Ratio), size = 3) + ggtitle("Percentage of
Flights Delayed by Destination") + labs(x="Destinations",y="Percentage")
```

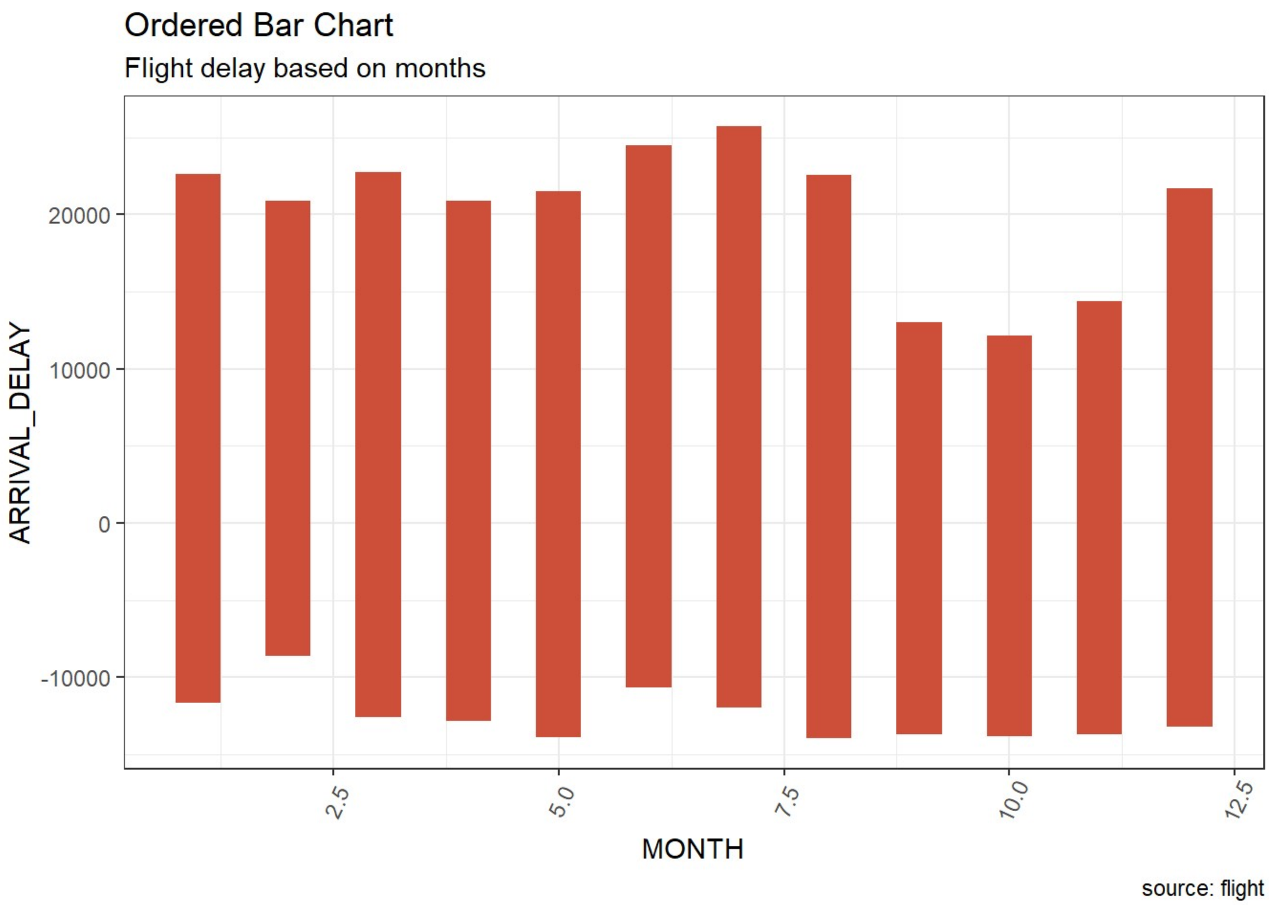
```
## Warning: Removed 4 rows containing missing values (geom_path).
```

```
## Warning: Removed 4 rows containing missing values (geom_text_repel).
```

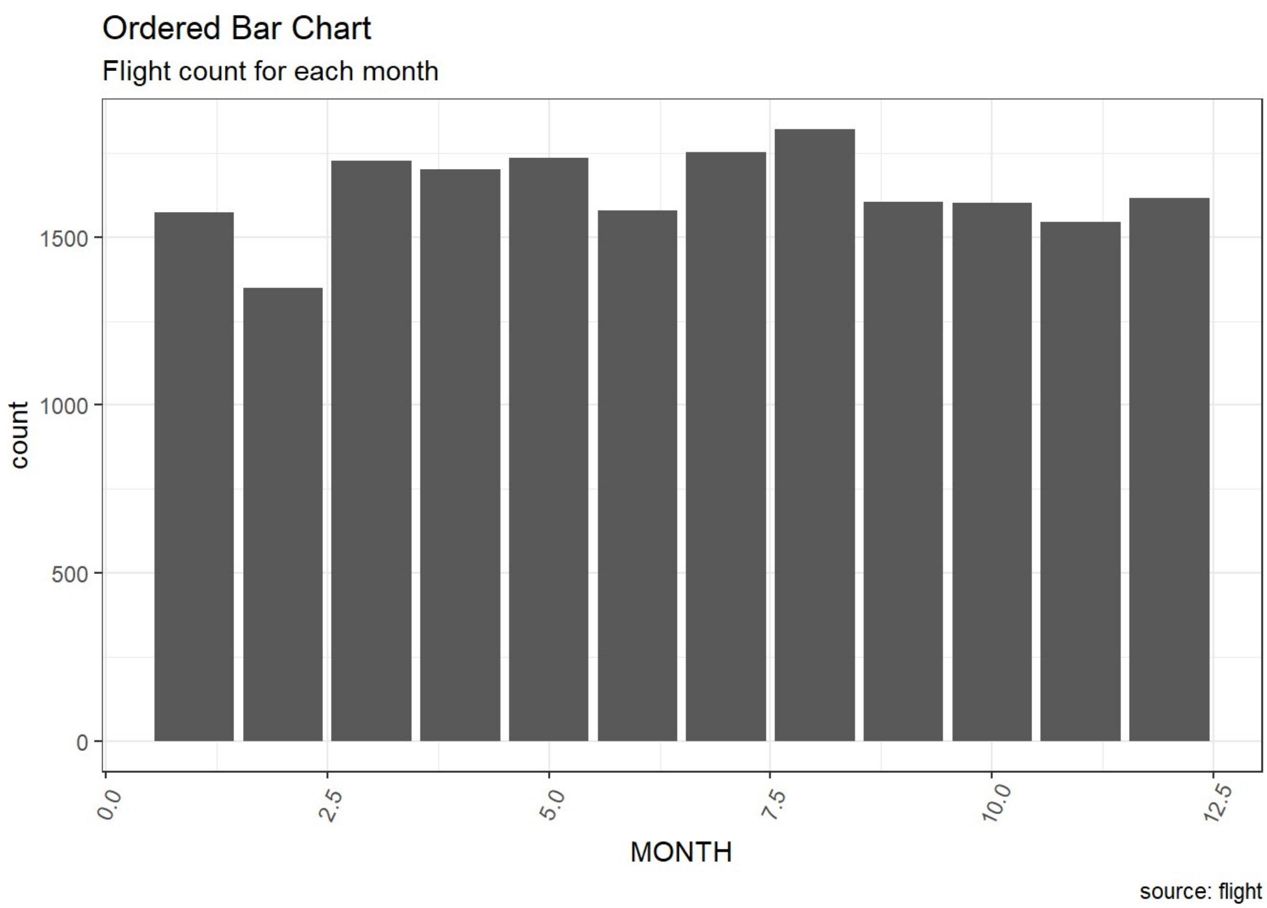



```
flight$CARRIER_CODE = as.numeric(as.factor(flight$AIRLINE))
flight$origin= as.numeric(as.factor(flight$ORIGIN_AIRPORT))
flight$dest=as.numeric(as.factor(flight$DESTINATION_AIRPORT))
flight$ARR_HOUR = floor(flight$ARRIVAL_TIME/100)
```

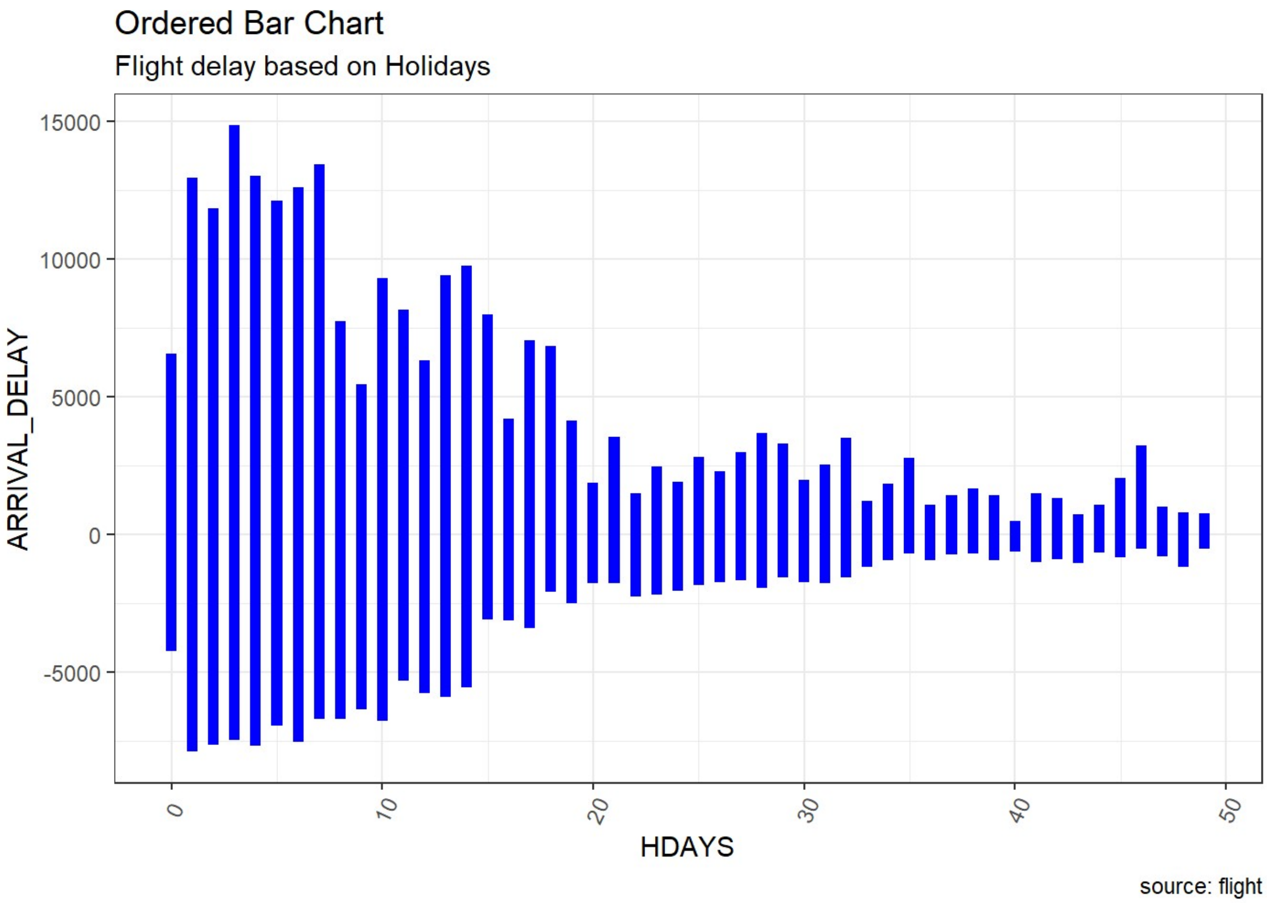
```r
# Create break points and labels for axis ticks
theme_set(theme_bw())

# Draw plot
ggplot(flight, aes(MONTH, ARRIVAL_DELAY)) +
  geom_bar(stat="identity", width=.5, fill="tomato3") +
  labs(title="Ordered Bar Chart",
       subtitle="Flight delay based on months",
       caption="source: flight") +
  theme(axis.text.x = element_text(angle=65, vjust=0.6))
```




```r
ggplot(flight, aes(x = MONTH)) + geom_bar() +labs(title="Ordered Bar Chart",
       subtitle="Flight count for each month",
       caption="source: flight") +
  theme(axis.text.x = element_text(angle=65, vjust=0.6))
```




```
#plot of delay as a result of proximity to holidays
ggplot(flight, aes(HDAYS, ARRIVAL_DELAY)) +
  geom_bar(stat="identity", width=.5, fill="blue") +
  labs(title="Ordered Bar Chart",
       subtitle="Flight delay based on Holidays",
       caption="source: flight") +
  theme(axis.text.x = element_text(angle=65, vjust=0.6))
```


Ordered Bar Chart
Flight delay based on Holidays
15000
10000
5000
0
-5000
ARRIVAL_DELAY
0
10
20
30
40
50
HDAYS
source: flight