# HW2

Vivian Chu

August 15, 2015

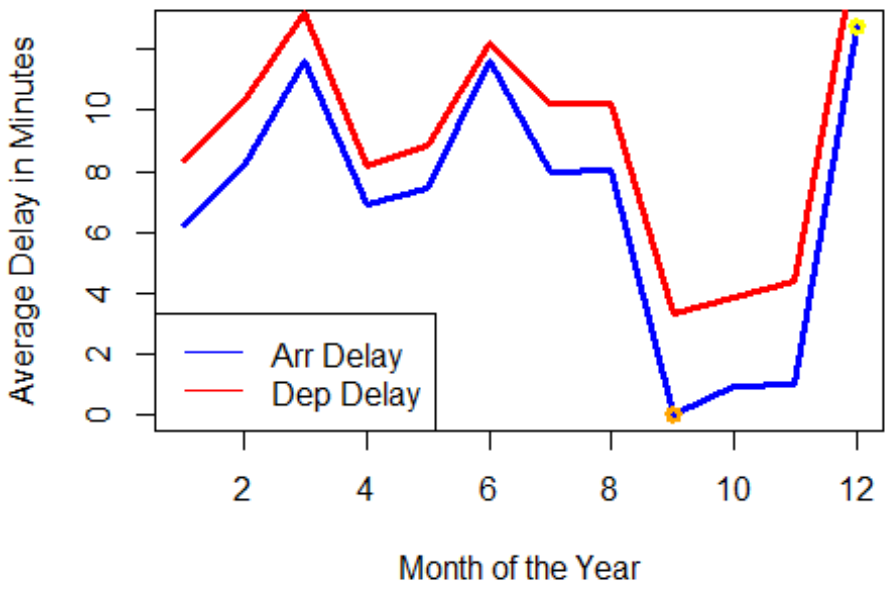
## Flights at ABIA

Questions:

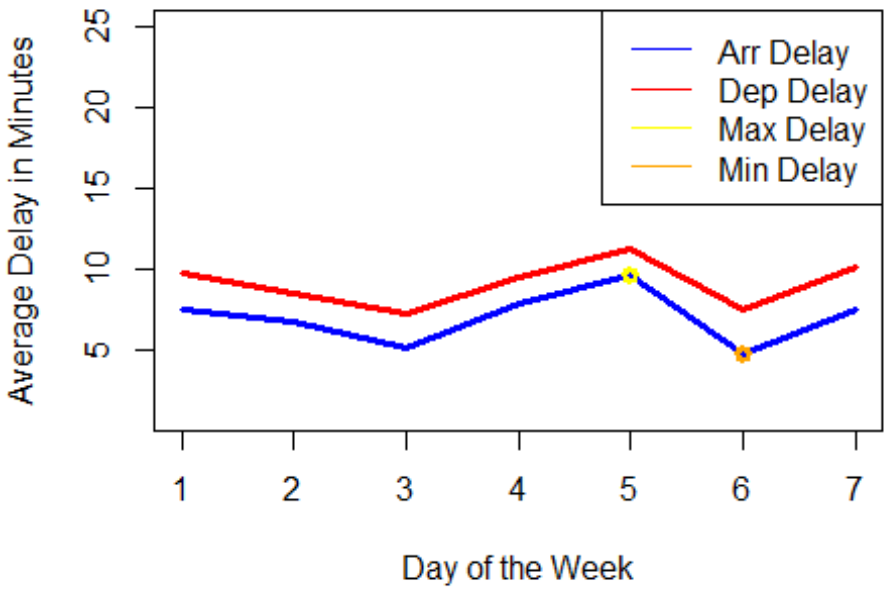
What day in the week is the worst time to fly? (Most amount of delays) What month in the year is the worst time to fly

```
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

flights =
read.csv('https://raw.githubusercontent.com/jgscott/STA380/master/data/
ABIA.csv')
flight = flights
# turn days of the week and months in a year into factors
df = flight %>%
  mutate(
    day_of_week=factor(DayOfWeek,levels=c(1,2,3,4,5,6,7),
                       labels=
                         c("Monday",
                           "Tuesday",
                           "Wednesday",
                           "Thursday",
                           "Friday",
                           "Saturday",
                           "Sunday"
                         )))
df = df %>%
  mutate(
    Month=factor(Month,levels=c(1,2,3,4,5,6,7,8,9,10,11,12),
                       labels=
```

```
                          c("Jan",
                            "Feb",
                            "Mar",
                            "Apr",
                            "May",
                            "Jun",
                            "Jul",
                            "Aug",
                            "Sept",
                            "Oct",
                            "Nov",
                            "Dec"
                          )))
#create a line chart showing the average delay in minutes across the
months in a year, circling the worst time to fly in yellow and the best
time to fly in orange

aggdelay = aggregate(df$ArrDelay, by=list(df$Month), FUN=mean,
na.rm=TRUE)
plot(aggdelay$x, type = "l", xlab= "Month of the Year", ylab = "Average
Delay in Minutes",col ="blue", lwd = "3")
aggdepdelay = aggregate(df$DepDelay, by=list(df$Month), FUN=mean,
na.rm=TRUE)
lines(aggdepdelay$x, type="l",col ="red", lwd="3")
legend('bottomleft',c("Arr Delay","Dep Delay"),lty=1,
col=c("blue","red"))
points(which.max(aggdelay$x),max(aggdelay$x),lwd=3,col="yellow")
points(which.min(aggdelay$x),min(aggdelay$x),lwd=3,col="orange")
```




```
#create a line chart showing the average delay in minutes across the
days in the week, circling the worst time to fly in yellow and the best
time to fly in orange
aggdelay2 = aggregate(df$ArrDelay, by=list(df$day_of_week), FUN=mean,
na.rm=TRUE)
aggdepdelay2 = aggregate(df$DepDelay, by=list(df$day_of_week),FUN=mean,
na.rm=TRUE)
plot(aggdelay2$x, type="l", xlab="Day of the Week", ylab="Average Delay
in Minutes", ylim=c(1,25),col="blue", lwd="3")
lines(aggdepdelay2$x, type="l", col="red", lwd="3")
legend('topright',c("Arr Delay","Dep Delay","Max Delay","Min
Delay"),lty=1, col=c("blue","red","yellow","orange"))
points(which.max(aggdelay2$x),max(aggdelay2$x),lwd=3,col="yellow")
points(which.min(aggdelay2$x),min(aggdelay2$x),lwd=3,col="orange")
```




Departure delay follows arrival delays, as expected, since if the plane arrives late, it will depart late, as the passengers can't leave if the plane hasn't come in yet. As expected, December has the most delays in the year as that's when most people take their vacations for Christmas. September has the least delays in the year, presumably because that's when school season starts and neither kids nor parents can get away. Friday has the max delays in the week as people are most likely to travel on Friday for a weekend away.

Busy airports lead to more possibilities of late passengers that have already checked their bags, and the plane cannot leave without the passenger if the bag is already checked. More planes flying in and out also means more planes queing for the landing field.

```
library(randomForest)
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.2.2
```

```
library(rpart)
library(ggplot2)
library(tm)
```

```
## Loading required package: NLP
##
## Attaching package: 'NLP'
##
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.2.2
```

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

```
#Read in the reader's function from tm_examples here
readerPlain = function(fname){
  readPlain(elem=list(content=readLines(fname)),
            id=fname, language='en') }
```

Create training set corpus

```
#Training set
author_dirs.tr = Sys.glob('../data/ReutersC50/C50train/*')
file_list = NULL
labels = NULL
for(author in author_dirs.tr) {
  author_name = substring(author, first=29)
  files_to_add = Sys.glob(paste0(author, '/*.txt'))
  file_list = append(file_list, files_to_add)
  labels = append(labels, rep(author_name, length(files_to_add)))
}
#Preprocessing
all_docs = lapply(file_list, readerPlain)
names(all_docs) = file_list
names(all_docs) = sub('.txt', '', names(all_docs))

tr.corpus = Corpus(VectorSource(all_docs))
names(tr.corpus) = file_list

tr.corpus = tm_map(tr.corpus, content_transformer(tolower))
tr.corpus = tm_map(tr.corpus, content_transformer(removeNumbers))
tr.corpus = tm_map(tr.corpus, content_transformer(removePunctuation))
tr.corpus = tm_map(tr.corpus, content_transformer(stripWhitespace))
tr.corpus = tm_map(tr.corpus, content_transformer(removeWords),
stopwords("SMART"))

#Forming document term matrix and dense matrix
```

```
tr.DTM = DocumentTermMatrix(tr.corpus)
tr.DTM = removeSparseTerms(tr.DTM, 0.975)
```

Create a test set corpus

```
#Test set
author_dirs.ts = Sys.glob('../data/ReutersC50/C50test/*')
file_list = NULL
labels.ts = NULL
for(author in author_dirs.tr) {
  author_name = substring(author, first=28)
  files_to_add = Sys.glob(paste0(author, '/*.txt'))
  file_list = append(file_list, files_to_add)
  labels = append(labels, rep(author_name, length(files_to_add)))
}
#Preprocessing
all_docs = lapply(file_list, readerPlain)
names(all_docs) = file_list
names(all_docs) = sub('.txt', '', names(all_docs))

ts.corpus = Corpus(VectorSource(all_docs))
names(ts.corpus) = file_list

ts.corpus = tm_map(ts.corpus, content_transformer(tolower))
ts.corpus = tm_map(ts.corpus, content_transformer(removeNumbers))
ts.corpus = tm_map(ts.corpus, content_transformer(removePunctuation))
ts.corpus = tm_map(ts.corpus, content_transformer(stripWhitespace))
ts.corpus = tm_map(ts.corpus, content_transformer(removeWords),
stopwords("SMART"))

#Extract terms from training set corpus
dict =dimnames(tr.DTM)[[2]]

#Forming document term matrix and dense matrix with only words in the
dictionary from training set

ts.DTM = DocumentTermMatrix(ts.corpus, list(dictionary=dict))
ts.DTM = removeSparseTerms(ts.DTM, 0.975)
```