678midterm

Xinyi Wang

11/17/2018

This is just a draft, will modify and polish after get some feedback

Read & Clean Data

```r
setwd("/Users/CindyWang/Desktop/678/midterm_project")
flights <- read.csv("flights.csv")
weather <- read.csv("weather.csv")
airlines <- read.csv("airlines.csv")
flights2 <- read.csv("flights2.csv")
weather2 <- read.csv("weather2.csv")
rank <- read.csv("rank.csv")

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
```

```r
train_weather <- weather %>%
  dplyr::select(NAME,DATE,WT01,WT02,WT03,WT05,WT08,SNOW,AWND,TAVG,PRCP) %>%
  filter(NAME=="ATLANTA HARTSFIELD INTERNATIONAL AIRPORT, GA US") %>%
  mutate(DAY_OF_MONTH = 1:31)
train_weather[is.na(train_weather)] <- 0

test_weather <- weather2 %>%
  dplyr::select(NAME,DATE,WT01,WT02,WT03,WT05,WT08,SNOW,AWND,TAVG,PRCP) %>%
  filter(NAME=="ATLANTA HARTSFIELD INTERNATIONAL AIRPORT, GA US") %>%
  mutate(DAY_OF_MONTH = 1:31)
test_weather[is.na(test_weather)] <- 0

#TRAIN
##join "flights" and "weather"
flights$X <- NULL
train <- inner_join(flights,train_weather,by="DAY_OF_MONTH")

##join "train" and "airlines" -> train
names(airlines)[names(airlines) == "Code"] <- "OP_UNIQUE_CARRIER"
train <- inner_join(train,airlines,by="OP_UNIQUE_CARRIER")
```

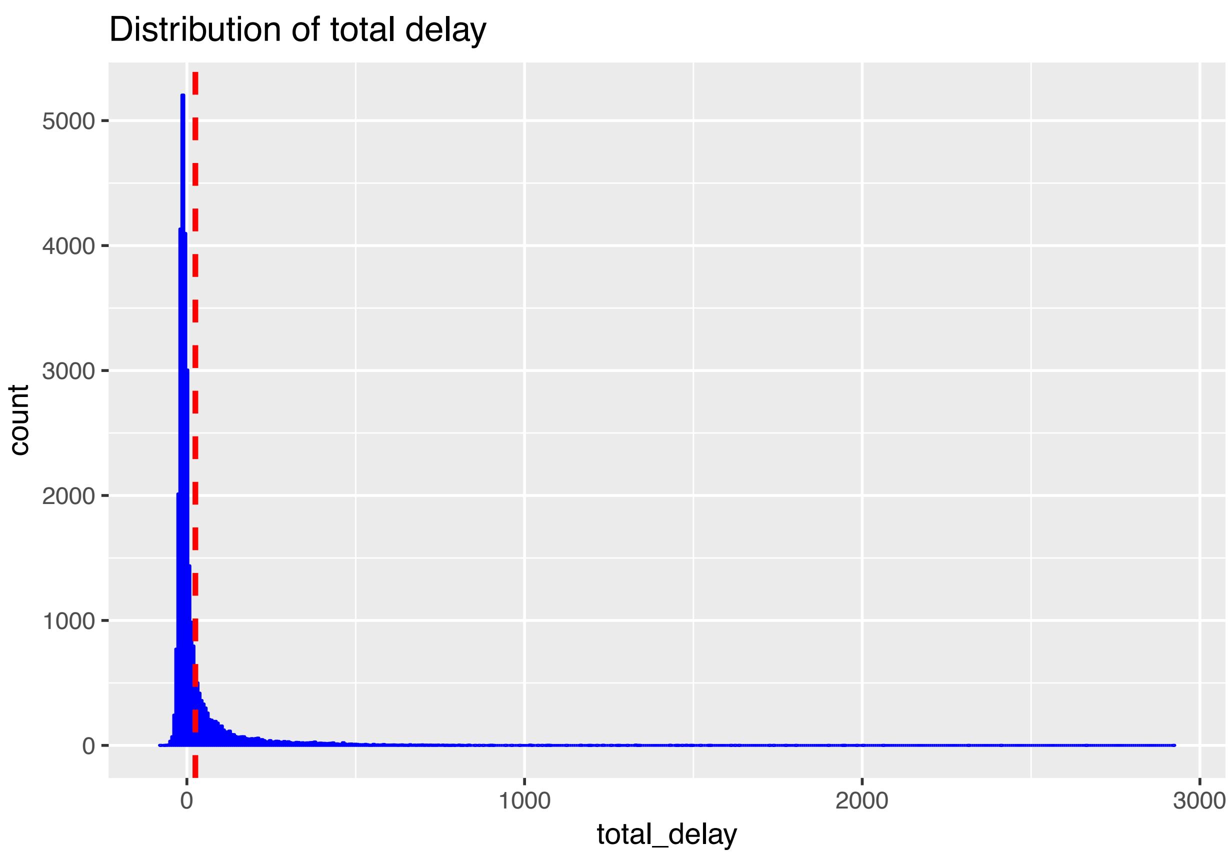
```
##join "train" and "rank"
names(rank)[names(rank) == "Code"] <- "OP_UNIQUE_CARRIER"
rank <- rank %>% dplyr::select(OP_UNIQUE_CARRIER,Rank)
train <- inner_join(train,rank,by="OP_UNIQUE_CARRIER")

##change the data class of the filtered data to enable data processing and running algorithms
train$DAY_OF_MONTH <- as.factor(train$DAY_OF_MONTH)
train$DAY_OF_WEEK <- as.factor(train$DAY_OF_WEEK)
# train$DEP_TIME_BLK <- as.factor(train$DEP_TIME_BLK)
train$ORIGIN <- as.character(train$ORIGIN)
train$DEST_STATE_ABR <- as.character(train$DEST_STATE_ABR)

#TEST
##join "flights2" and "weather2"
flights2$X <- NULL
test <- inner_join(flights2,test_weather,by="DAY_OF_MONTH")

##join "test" and "airlines" -> test
names(airlines)[names(airlines) == "Code"] <- "OP_UNIQUE_CARRIER"
test <- inner_join(test,airlines,by="OP_UNIQUE_CARRIER")

##join "train" and "rank"
test <- inner_join(test,rank,by="OP_UNIQUE_CARRIER")

##change the data class of the filtered data to enable data processing and running algorithms
test$DAY_OF_MONTH <- as.factor(test$DAY_OF_MONTH)
test$DAY_OF_WEEK <- as.factor(test$DAY_OF_WEEK)
# train$DEP_TIME_BLK <- as.factor(train$DEP_TIME_BLK)
test$ORIGIN <- as.character(test$ORIGIN)
test$DEST_STATE_ABR <- as.character(test$DEST_STATE_ABR)

##Clean "train"
train$YEAR <- NULL
train$DEP_DELAY_NEW <- NULL
train$ARR_DELAY_NEW <- NULL
train$MONTH <- NULL
train$TAXI_IN <- NULL
train$TAXI_OUT <- NULL
train$WHEELS_ON <- NULL
train$WHEELS_OFF <- NULL
train$CANCELLED <- NULL

##Clean "test"
test$YEAR <- NULL
test$DEP_DELAY_NEW <- NULL
test$ARR_DELAY_NEW <- NULL
test$MONTH <- NULL
test$TAXI_IN <- NULL
test$TAXI_OUT <- NULL
test$WHEELS_ON <- NULL
test$WHEELS_OFF <- NULL
test$CANCELLED <- NULL
```

```
train <- train %>%
  filter(ORIGIN=="ATL") %>%
  mutate(total_delay=DEP_DELAY+ARR_DELAY) %>%
  na.omit()

test <- test %>%
  filter(ORIGIN=="ATL") %>%
  mutate(total_delay=DEP_DELAY+ARR_DELAY) %>%
  na.omit()
```

EDA

1.This figure shows that the distribution of the outcome is right skewed, it has long tail in the high values. And we can see from the plot that most of flights are delay less than 25min but there are some outliers(flight delay alomost 3000min).

```
library(ggplot2)
ggplot(data = train, aes(x=total_delay)) +
  geom_histogram(color="blue", bins = 500) +
  geom_vline(aes(xintercept=mean(total_delay)),
             color="red", linetype="dashed", size=1) +
  labs(title="Distribution of total delay") +
  theme_gray()
```



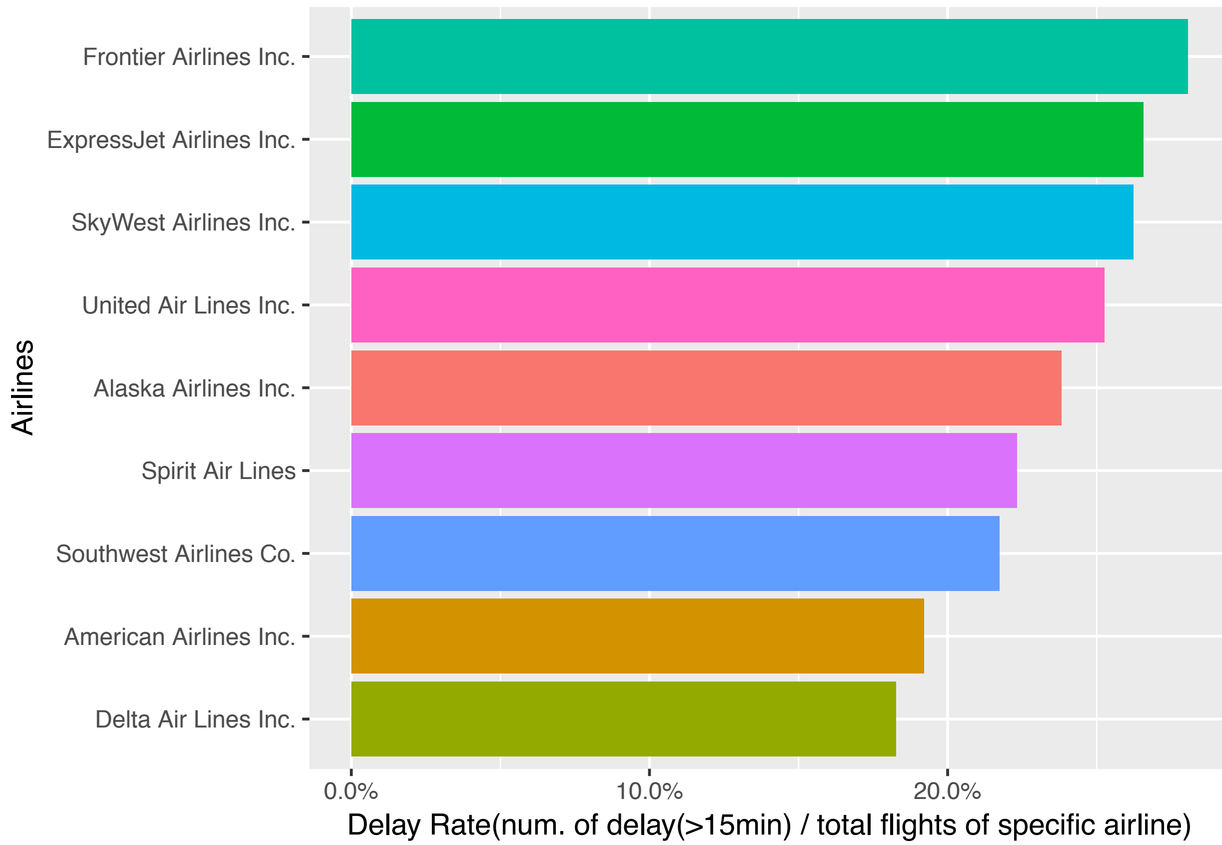

```
mean(train$total_delay)
```

```
## [1] 25.10688
```

```r
# train$delay_pos<-train$total_delay
# train$delay_pos[train$total_delay<=0]<-0
#
# ggplot(data = train, aes(x=DAY_OF_MONTH,y=log10(delay_pos/60),color=factor(OP_UNIQUE_CARRIER))) +
#   geom_point(alpha=0.1) +
#   labs(title="Distribution of total delay") +
#   geom_smooth(method="lm",se=FALSE)+ theme(legend.position="none")+
#   theme_gray()
#
#  geom_vline(aes(xintercept=mean(total_delay)),
#             color="red", linetype="dashed", size=1) +


# library(ggplot2)
# ggplot(data = train[train$total_delay>0,], aes(x=(total_delay/60)^(-1/5))) +
#   geom_histogram(color="blue", bins = 500) +
#   geom_vline(aes(xintercept=mean(total_delay)),
#             color="red", linetype="dashed", size=1) +
#   labs(title="Distribution of total delay")
```

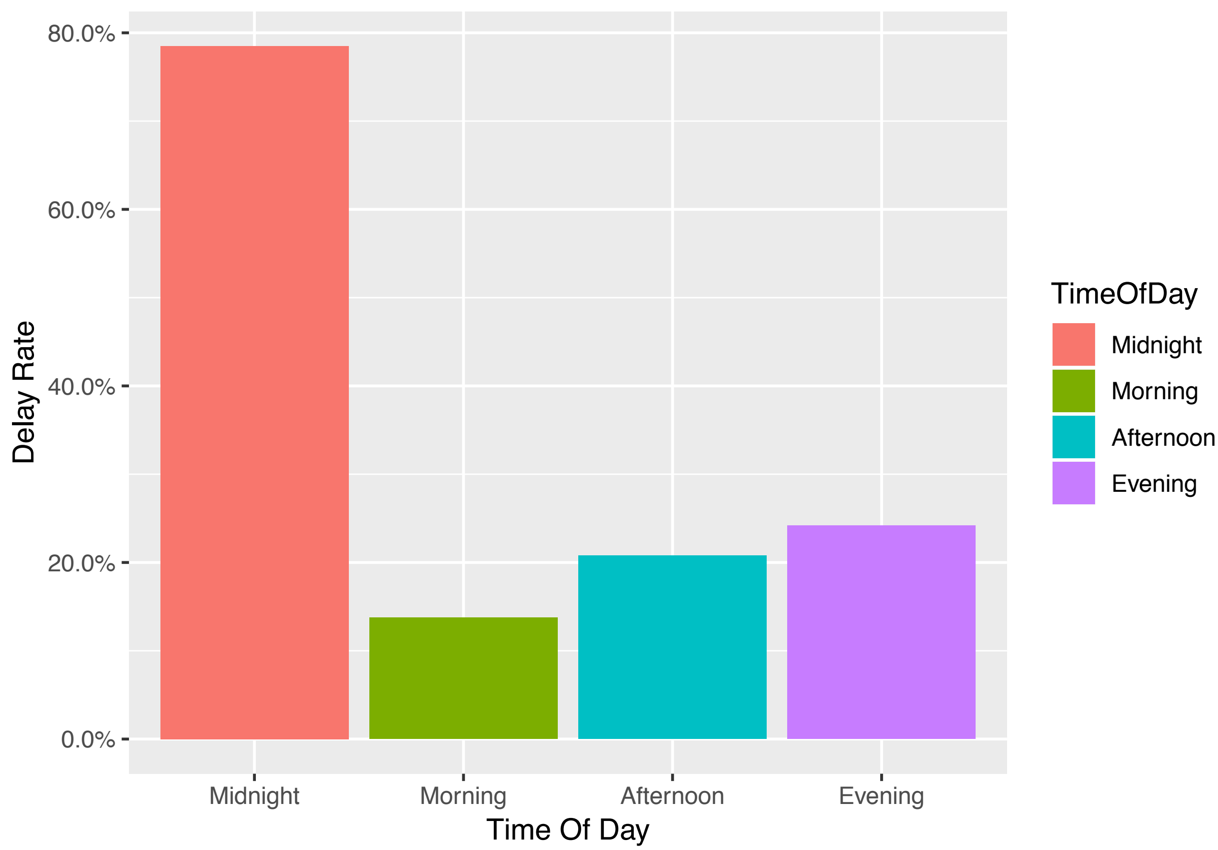
2.

```r
carrier_count <- train %>%
  dplyr::select(Description,ARR_DEL15) %>%
  group_by(Description) %>%
  summarise(total=n(),delay=sum(ARR_DEL15==1),percentage=(delay/total))

carrier_bar <- ggplot(carrier_count,aes(x=reorder(Description,percentage),
                                        y=percentage,fill=Description)) +
  geom_bar(stat="identity") +
  xlab("Airlines") +
  ylab("Delay Rate(num. of delay(>15min) / total flights of specific airline)") +
  scale_y_continuous(labels = scales::percent) +
  coord_flip() +
  theme_gray() +
  theme(legend.position="none")


carrier_bar
```



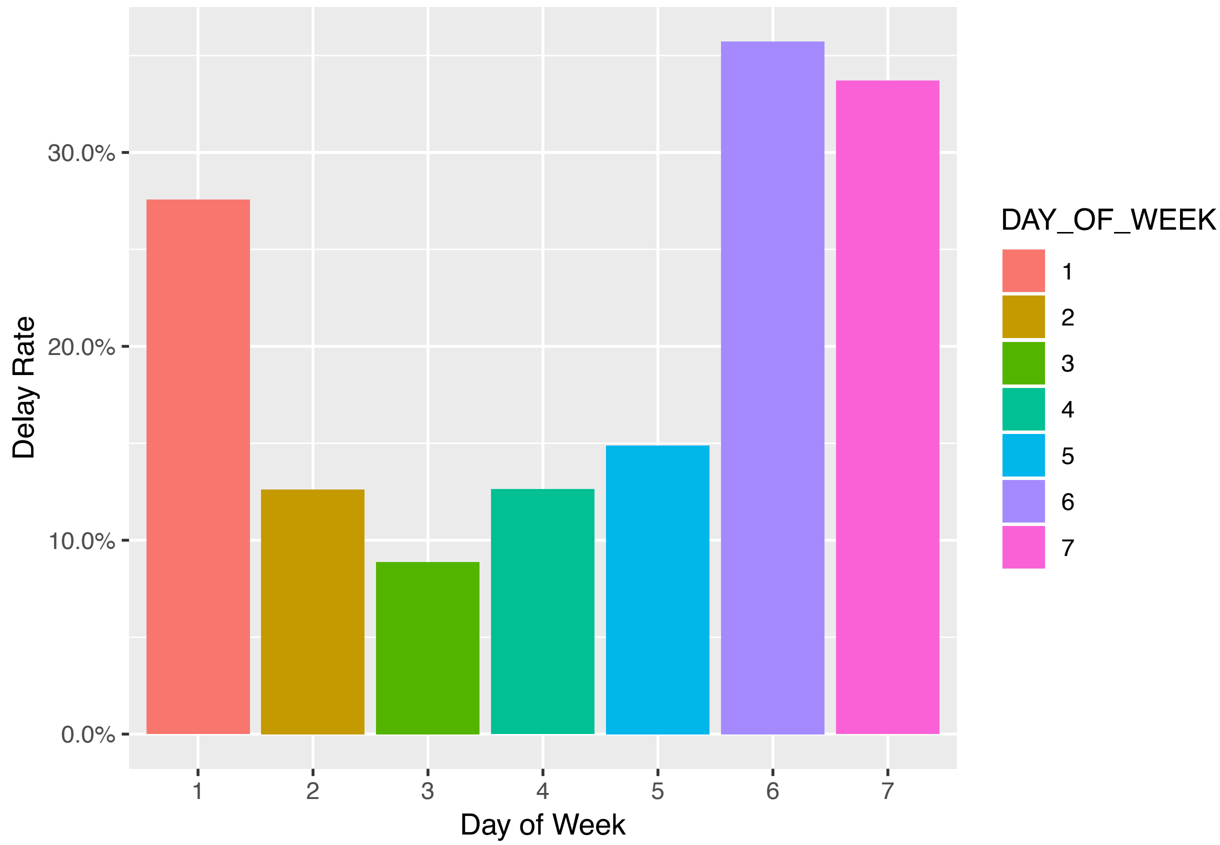
3.

```
train <- train %>%
  mutate(TimeOfDay = cut(DEP_TIME, c(0, 600, 1200, 1800, 2400),
                         labels = c("Midnight", "Morning", "Afternoon","Evening"), right = TRUE))

timeofday_count <- train %>%
  dplyr::select(TimeOfDay,ARR_DEL15) %>%
  group_by(TimeOfDay) %>%
  summarise(total=n(),delay=sum(ARR_DEL15==1),percentage=(delay/total))

timeofday_bar <- ggplot(timeofday_count,aes(x=TimeOfDay,y=percentage,fill=TimeOfDay)) +
  geom_bar(stat="identity") +
  xlab("Time Of Day") +
  ylab("Delay Rate") +
  scale_y_continuous(labels = scales::percent) +
  theme_gray()
timeofday_bar
```



4.

```
dayofweek_count <- train %>%
  dplyr::select(DAY_OF_WEEK,ARR_DEL15) %>%
  group_by(DAY_OF_WEEK) %>%
  summarise(total=n(),delay=sum(ARR_DEL15==1),percentage=(delay/total))

dayofweek_bar <- ggplot(dayofweek_count,aes(x=DAY_OF_WEEK,y=percentage,fill=DAY_OF_WEEK)) +
  geom_bar(stat="identity") +
  xlab("Day of Week") +
  ylab("Delay Rate") +
  scale_y_continuous(labels = scales::percent) +
  theme_gray()
dayofweek_bar
```



5.This figure shows scatter plots of the predictors against the outcome along with a regression line from a flexible "smoother" model. According to these two figures, we can assume that the relationship between the predictors and the outcome is linear.

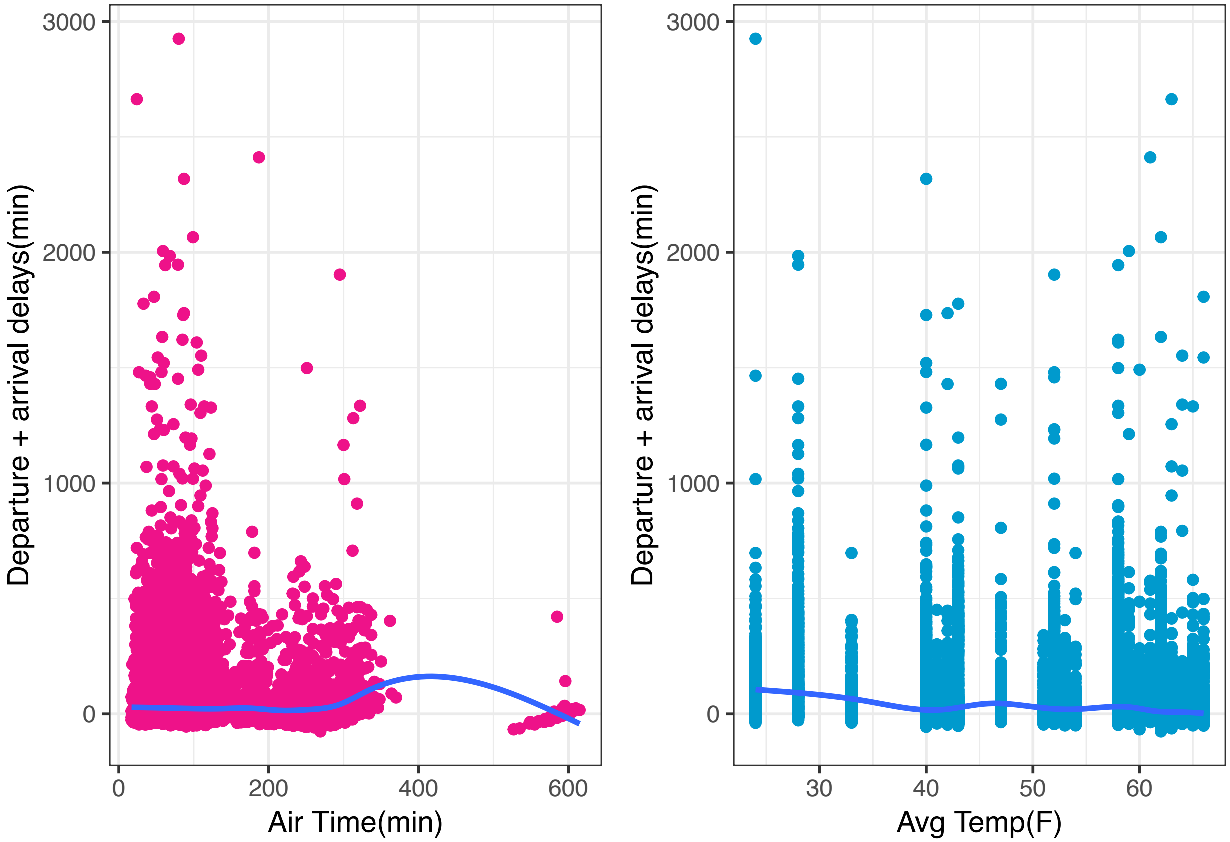
```
library(cowplot) #Arranging plots in a grid
```

```
##
## Attaching package: 'cowplot'

## The following object is masked from 'package:ggplot2':
##
##     ggsave
```

```
fig1 <- ggplot(data=train, aes(x = AIR_TIME, y = total_delay)) +
  geom_point(color ="deeppink2")+
  geom_smooth(se=F)+
  labs( x="Air Time(min)", y="Departure + arrival delays(min)")+
  theme_bw()

fig2 <- ggplot(data=train, aes(x = TAVG, y = total_delay)) +
  geom_point(color ="deepskyblue3")+
  geom_smooth(se=F)+
  labs( x="Avg Temp(F)", y="Departure + arrival delays(min)")+
  theme_bw()

plot_grid(fig1, fig2)
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

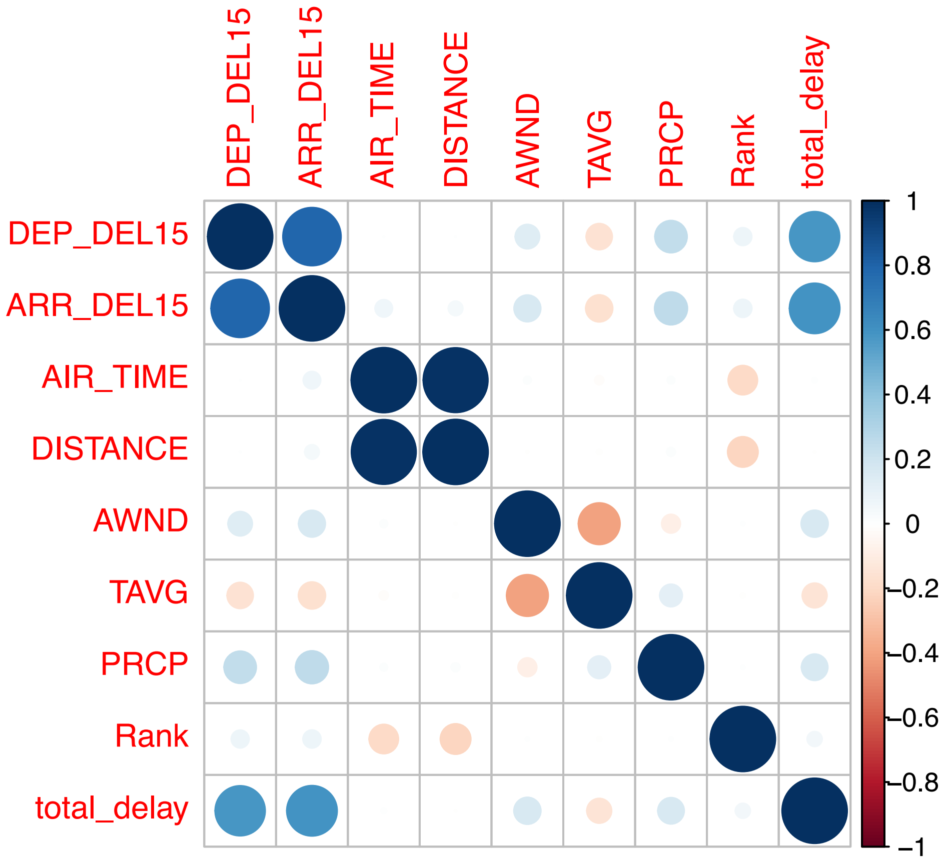


6. correlation

```
train$WT01 <- as.factor(train$WT01)
train$WT02 <- as.factor(train$WT02)
train$WT03 <- as.factor(train$WT03)
train$WT05 <- as.factor(train$WT05)
train$WT08 <- as.factor(train$WT08)
train$SNOW <- as.numeric(train$SNOW)
data.num <- Filter(is.numeric, train)
data.num$DEP_TIME <- NULL
data.num$DEP_DELAY <- NULL
data.num$DEP_DELAY_GROUP <- NULL
data.num$ARR_TIME <- NULL
data.num$ARR_DELAY <- NULL
data.num$ARR_DELAY_GROUP <- NULL
data.num$SNOW <- NULL
correlations <- cor(data.num)
# dim(correlations)
# print(correlations)

library(corrplot)
```

```
## corrplot 0.84 loaded
```

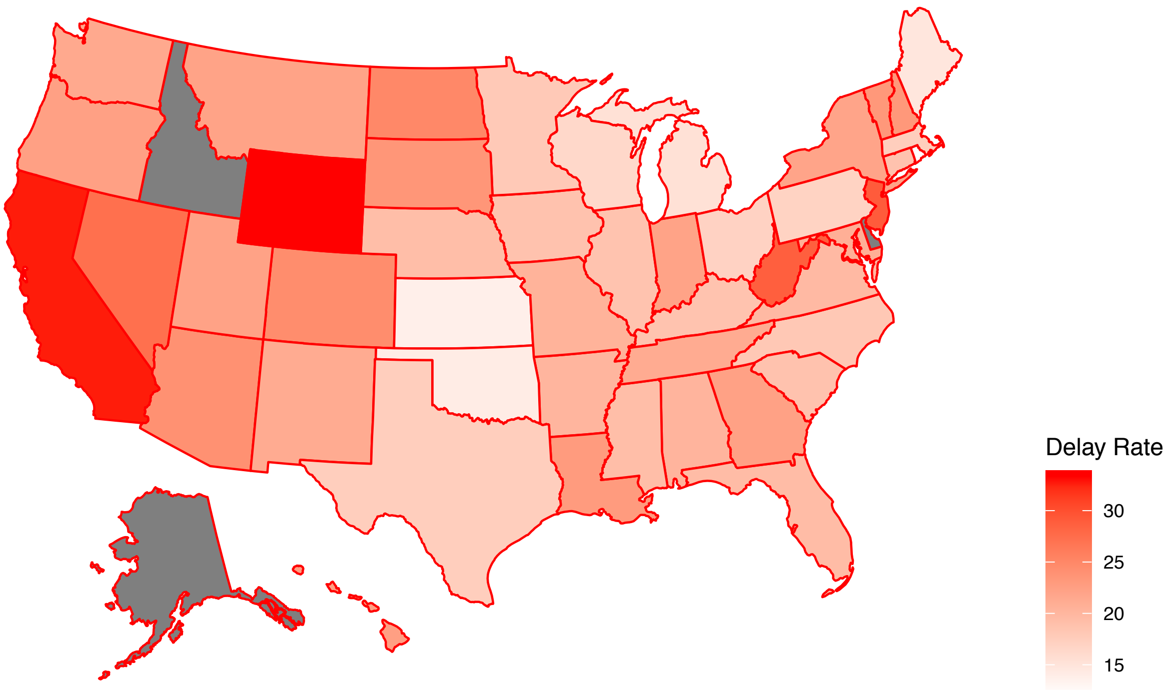
```
corrplot(correlations,method="circle")
```




```
##The variables "DEP_DEL15" and "AIR_TIME" are highly correlated with others predictors.
# train <- train %>% select(-DEP_DEL15,-AIR_TIME)
```

7.U.S heat map

```
library(usmap)
library(ggplot2)

dest_count <- train %>%
  dplyr::select(DEST_STATE_ABR,ARR_DEL15) %>%
  group_by(DEST_STATE_ABR) %>%
  summarise(total=n(),delay=sum(ARR_DEL15==1),percentage=(delay/total)*100)
dest_count <- as.data.frame(dest_count)
names(dest_count)[names(dest_count) == 'DEST_STATE_ABR'] <- 'state'

plot_usmap(data = dest_count, values = "percentage", lines = "red") +
  scale_fill_continuous(low = "white", high = "red",name = "Delay Rate",
                        label = scales::comma) +
  theme(legend.position = "right")
```



Feature Selection

```
# str(train)
##change the data class of the filtered data to enable data processing and running algorithms
train$OP_UNIQUE_CARRIER <- as.factor(train$OP_UNIQUE_CARRIER)
train$DEST_STATE_ABR <- as.factor(train$DEST_STATE_ABR)
train$ARR_DEL15 <- as.factor(train$ARR_DEL15)
train$WT01 <- as.factor(train$WT01)
train$WT02 <- as.factor(train$WT02)
train$WT05 <- as.factor(train$WT05)
train$WT08 <- as.factor(train$WT08)
train$DEST <- as.factor(train$DEST)
train$Rank <- as.factor(train$Rank)

##Clean
train$ORIGIN <- NULL
train$ORIGIN_CITY_NAME <- NULL
train$ORIGIN_STATE_ABR <- NULL
train$DEST_CITY_NAME <- NULL
train$DEP_TIME <- NULL
train$ARR_TIME <- NULL
train$NAME <- NULL
train$DATE <- NULL
train$Description <- NULL
train$WT03 <- NULL
train$WT05 <- NULL
train$WT08 <- NULL
train$DEP_DELAY_GROUP <- NULL
train$ARR_DELAY_GROUP <- NULL
```

```r
train$DEST <- NULL

# str(train)

#try 1(keep)
# library(Boruta)
# boruta_output <- Boruta(total_delay~., data = train, doTrace = 2)
# boruta_signif <- names(boruta_output$finalDecision[boruta_output$finalDecision %in% c("Confirmed", "T
# print(boruta_signif)
# plot(boruta_output, cex.axis=.5, las=2, xlab="", main="Variable Importance")
## Boruta performed 99 iterations in 6.7969 mins.
## 16 attributes confirmed important: AIR_TIME, ARR_DEL15, ARR_DELAY, AWND, DAY_OF_MONTH and 11 more;
## 2 attributes confirmed unimportant: DEST_STATE_ABR, SNOW;
## 1 tentative attributes left: WT03;

# train <- train %>% select(-DEST_STATE_ABR,-SNOW,-DEP_DELAY,-ARR_DELAY)
train <- train %>% dplyr::select(-DEST_STATE_ABR,-SNOW)

# #try 2
# base.mod <- lm(total_delay ~ 1 , data= train)  # base intercept only model
# all.mod <- lm(total_delay ~ . , data= train) # full model with all predictors
# stepMod <- step(base.mod, scope = list(lower = base.mod, upper = all.mod), direction = "forward", tra
# shortlistedVars <- names(unlist(stepMod[[1]])) # get the shortlisted variable.
# shortlistedVars <- shortlistedVars[!shortlistedVars %in% "(Intercept)"]  # remove intercept
# print(shortlistedVars)
#
# #try 3
# library(randomForest)
# str(train)
# rf=randomForest(total_delay ~ . , data = train)
```

test sample

```r
#use "ARR_DEL15" as y
test <- test %>%
  mutate(TimeOfDay = cut(DEP_TIME, c(0, 600, 1200, 1800, 2400),
                   labels = c("Midnight", "Morning", "Afternoon","Evening"), right = TRUE)) %>%
  dplyr::select(DAY_OF_MONTH,DAY_OF_WEEK,OP_UNIQUE_CARRIER,DEP_DEL15,ARR_DEL15,AIR_TIME,DISTANCE,WT01,W
  filter(!OP_UNIQUE_CARRIER %in% setdiff(test$OP_UNIQUE_CARRIER,train$OP_UNIQUE_CARRIER))


test$OP_UNIQUE_CARRIER <- as.factor(test$OP_UNIQUE_CARRIER)
test$ARR_DEL15 <- as.factor(test$ARR_DEL15)
test$DEP_DEL15 <- as.factor(test$DEP_DEL15)
test$WT01 <- as.factor(test$WT01)
test$WT02 <- as.factor(test$WT02)
test$DAY_OF_MONTH <- as.numeric(test$DAY_OF_MONTH)
train$DAY_OF_MONTH <- as.numeric(train$DAY_OF_MONTH)


#################
```

```r
# ##use "total_delay" as y
# test <- test %>%
#   mutate(TimeOfDay = cut(DEP_TIME, c(0, 600, 1200, 1800, 2400),
#                          labels = c("Midnight", "Morning", "Afternoon","Evening"), right = TRUE)) %>%
#   select(DAY_OF_MONTH,DAY_OF_WEEK,OP_UNIQUE_CARRIER,ARR_DEL15,DEP_DEL15,AIR_TIME,DISTANCE,WT01,WT02,A
#
# test$OP_UNIQUE_CARRIER <- as.factor(test$OP_UNIQUE_CARRIER)
# test$ARR_DEL15 <- as.factor(test$ARR_DEL15)
# test$DEP_DEL15 <- as.factor(test$DEP_DEL15)
# test$WT01 <- as.factor(test$WT01)
# test$WT02 <- as.factor(test$WT02)
# test$Rank <- as.factor(test$Rank)
#
# train <- train %>% select(-DEP_DELAY,-ARR_DELAY)
# train$DAY_OF_WEEK <- as.factor(train$DAY_OF_WEEK)
```

Model

```r
# str(train)

# ## linear model
# m1 <- lm(total_delay ~ DAY_OF_MONTH + DAY_OF_WEEK + OP_UNIQUE_CARRIER + DEP_DEL15 + ARR_DEL15 +
#             AIR_TIME  + WT01 + WT02 + AWND + TAVG + PRCP +
#             Rank + TimeOfDay, data = train)
# summary(m1)
#
# plot(m1, which = 1)
# par(mfrow= c(2,3))
# plot(m1, which = 1:6)
#
# ## multilevel model
# library(lme4)
# m2 <- lmer(scale(total_delay) ~ DAY_OF_MONTH + DAY_OF_WEEK + DEP_DEL15 + ARR_DEL15 +AIR_TIME + DISTAN
# summary(m2)


##logistic model
library(car)
```

```
## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode
```

```r
library(lme4)
```

```
## Loading required package: Matrix
```

```r
m3 <- glmer(ARR_DEL15 ~ DAY_OF_MONTH  + (1|OP_UNIQUE_CARRIER)  + WT01 + WT02 + AWND + TAVG + PRCP +
           DAY_OF_WEEK + TimeOfDay + DISTANCE, data = train,  family = binomial(link = "logit"))
summary(m3)
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula:
## ARR_DEL15 ~ DAY_OF_MONTH + (1 | OP_UNIQUE_CARRIER) + WT01 + WT02 +
##     AWND + TAVG + PRCP + DAY_OF_WEEK + TimeOfDay + DISTANCE
##    Data: train
##
##      AIC      BIC   logLik deviance df.resid
##  24491.3  24640.5 -12227.7  24455.3    29399
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.0855 -0.4547 -0.3179 -0.2025  7.7204
##
## Random effects:
##  Groups            Name        Variance Std.Dev.
##  OP_UNIQUE_CARRIER (Intercept) 0.07527  0.2744
## Number of obs: 29417, groups:  OP_UNIQUE_CARRIER, 9
##
## Fixed effects:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         8.898e-01  2.332e-01   3.815 0.000136 ***
## DAY_OF_MONTH       -3.610e-02  2.896e-03 -12.464  < 2e-16 ***
## WT011               1.606e-01  6.726e-02   2.387 0.016990 *
## WT021               9.443e-02  8.228e-02   1.148 0.251088
## AWND                1.442e-01  8.281e-03  17.410  < 2e-16 ***
## TAVG               -1.377e-02  2.957e-03  -4.659 3.18e-06 ***
## PRCP                5.917e-01  3.785e-02  15.634  < 2e-16 ***
## DAY_OF_WEEK2       -5.129e-01  6.316e-02  -8.120 4.65e-16 ***
## DAY_OF_WEEK3       -9.772e-01  9.631e-02 -10.147  < 2e-16 ***
## DAY_OF_WEEK4       -3.740e-01  6.443e-02  -5.805 6.42e-09 ***
## DAY_OF_WEEK5       -4.339e-01  6.218e-02  -6.977 3.01e-12 ***
## DAY_OF_WEEK6        1.705e-01  6.162e-02   2.767 0.005653 **
## DAY_OF_WEEK7        3.713e-01  5.788e-02   6.414 1.42e-10 ***
## TimeOfDayMorning   -3.063e+00  1.410e-01 -21.714  < 2e-16 ***
## TimeOfDayAfternoon -2.506e+00  1.393e-01 -17.990  < 2e-16 ***
## TimeOfDayEvening   -2.329e+00  1.398e-01 -16.658  < 2e-16 ***
## DISTANCE            4.388e-04  3.453e-05  12.710  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

##
## Correlation matrix not shown by default, as p = 17 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it

## convergence code: 0
## Model failed to converge with max|grad| = 0.0519894 (tol = 0.001, component 1)
## Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
```

```
## Model is nearly unidentifiable: large eigenvalue ratio
##  - Rescale variables?
```

```r
# marginalModelPlots(m3)

m3.predict <- predict(m3, test,type="response")
m3.predict <- ifelse(m3.predict > 0.5,1,0)
head(m3.predict)
```

```
## 1 2 3 4 5 6
## 0 0 0 0 1 0
```

```r
m3.predict <- as.factor(m3.predict)
compare <- data.frame(obs = test$ARR_DEL15, pred = m3.predict)

library(caret)
```

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

```r
confusionMatrix(m3.predict, test$ARR_DEL15)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 21585  3952
##          1  1164   425
##
##                Accuracy : 0.8114
##                  95% CI : (0.8067, 0.816)
##     No Information Rate : 0.8386
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0618
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9488
##             Specificity : 0.0971
##          Pos Pred Value : 0.8452
##          Neg Pred Value : 0.2675
##              Prevalence : 0.8386
##          Detection Rate : 0.7957
##    Detection Prevalence : 0.9414
##       Balanced Accuracy : 0.5230
##
##        'Positive' Class : 0
##
```

```r
#Accuracy:81.14%

library(arm)
```

```
## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
```

```
##
##     select

##
## arm (Version 1.10-1, built: 2018-4-12)

## Working directory is /Users/CindyWang/Desktop/678/midterm_project

##
## Attaching package: 'arm'

## The following object is masked from 'package:car':
##
##     logit

## The following object is masked from 'package:corrplot':
##
##     corrplot
```

```
binnedplot(fitted(m3),residuals(m3,type="response"))
```